A TECHNIQUE FOR MONITORING A REAL-TIME MULTI-COMPUTER SYSTEM UNDER HEAVY LOADING

by

Edward Allen Potter

United States
Naval Postgraduate School

THESIS

A TECHNIQUE FOR MONITORING
A REAL-TIME MULTI-COMPUTER SYSTEM
UNDER HEAVY LOADING

by

Edward Allen Potter

December 1970

A Technique For Monitoring
A Real-Time Multi-Computer System
Under Heavy Loading

by

Edward Allen Potter
Lieutenant, United States Navy
BSEE, University of Southern California, 1965

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL

ABSTRACT

Techniques are developed for the design of a monitor of a real-time multi-computer system that is under heavy loading. The first portion relates to the requirements of partitioning to aid in fault recognition and diagnostic routines. The dynamic allocation of system time to the system tasks and fault monitoring is developed secondly. System reconfiguration of the partitioned subsystems restores the system to operation at a degraded level until faults are corrected. The paper discusses a Ship Combat Weapon System as an example of a large scale multi-computer system monitor.

TABLE OF CONTENTS

I. INTRODUCTION ---------- 5

II. BASIC ELEMENTS OF A FAULT MONITORING SYSTEM ---------- 7

III. DEFINITIONS AND CONCEPTS OF FAULT MONITORING IN A MULTI-COMPUTER SYSTEM ---------- 10

A. PARTITIONING ---------- 10

B. DYNAMIC TIME ALLOCATION ---------- 12

C. FAULT RECOGNITION ---------- 13

D. DIAGNOSTIC ROUTINES ---------- 14

E. SYSTEM RECONFIGURATION ---------- 15

IV. A SPECIFIC EXAMPLE: A SHIP COMBAT WEAPON SYSTEM ---------- 17

A. PROPERTIES OF A COMBAT WEAPON SYSTEM AND THE DATA INVOLVED ---------- 18

B. THE SIMULATED COMBAT WEAPON SYSTEM ---------- 20

1. Types of Data Involved ---------- 22

2. Partitioning ---------- 28

a. Hardware ---------- 28

b. Software ---------- 29

3. Interface Requirements ---------- 29

a. Data Paths ---------- 29

b. A Specific Example ---------- 35

4. Monitor Program Module ---------- 35

a. Timing Analysis ---------- 35

(1) Light loading mode ---------- 37

(2) Medium loading ---------- 37

(3) Heavy loading ---------- 37

b. Data Sampling ---------- 38

c. Human Operator Interface - - - - - - - - - - - - - - 40

5. Combat Information Center Program (SMULA) - - - - - - 42

a. Main Control and Combat Information Display (CICP) - 42

b. Radar and Ship Information Simulator (RADAR) - - - 43

6. Fire Control System Program (Missile Mount) - - - - - 44

a. Digital Fire Control System - - - - - - - - - - - - 45

b. Missile Launcher Mount - - - - - - - - - - - - - - - 45

V. RECOMMENDED TECHNIQUES - 47

A. DYNAMIC TIME ALLOCATION - - - - - - - - - - - - - - - - - 47

B. PARTITIONING - 48

C. FAULT RECOGNITION - 49

D. DIAGNOSTIC ROUTINES - - - - - - - - - - - - - - - - - - - 49

E. SYSTEM RECONFIGURATION AND PRESENTATION - - - - - - - - - 50

COMPUTER PROGRAM A. Main Control and Combat Information Display (CICP) - - - - - - - - - - - - - - - - 53

B. Radar and Ship Information Simulator (RADAR) -109

C. The Digital Fire Control System - - - - - - - -144

D. The Missile Mount - - - - - - - - - - - - - - - -152

LIST OF REFERENCED ARTICLES -153

BIBLIOGRAPHY -155

A. Complete List of Articles - - - - - - - - - - - - - - - - - -155

B. Software Related Article Cross Reference - - - - - - - - - -160

C. Hardware Related Article Cross Reference - - - - - - - - - -161

INITIAL DISTRIBUTION LIST -162

FORM DD 1473 -163

I. INTRODUCTION

The monitoring of a small computer system is a relatively simple task. But as computer systems grow larger and larger, the problem becomes much greater. When the monitoring of a large scale Real-Time Multi-Computer system is attacked, many problems arise. One must know which program is running in which computer and how each program affects the mass of data that flows between the many input/output ports of the system. When the system is running near capacity in all processors (heavily loaded), the problem becomes even greater. There is little time left to process data for monitoring. Therefore, it can be seen that monitoring must be done under very severe timing and space utilization constraints. This thesis provides the necessary tools for calculating timing and core space requirements.

The five major portions of a fault monitoring system are (1) Partitioning of the total system into small subsystems, (2) Dynamic time allocation to better utilize remaining time for fault monitoring procedures, (3) Fault recognition techniques, (4) Diagnostic routines, (5) System reconfiguration to automatically restore system operation. All five of these operations are fully discussed and analyzed so that the designer may apply the correct techniques to his system to obtain an effective monitor.

One example of a Real-Time Multi-Computer system is a Ship Combat Weapon System. The general aspects of a Ship Combat Weapon System are discussed to show the comparisons of a general system to a specific one. A simulated Ship Combat Weapon System is shown to contain all

the necessary parts of the general system and is analyzed in great detail. The methods of monitoring this simulated Ship Combat Weapon System are analyzed to develop the necessary monitoring techniques.

A detailed bibliography is presented that covers the area of Monitoring of a Real-Time Multi-Computer System. A cross reference of the methods of monitoring is also given.

Proper system time utilization shows that much of the system upkeep and maintenance that normally is accomplished during designated maintenance periods may now be performed on-line at little or no system degradation.

II. BASIC ELEMENTS OF A FAULT MONITORING SYSTEM

Fault detection and recognition is the most important maintenance function in a Real-Time System. The Fault Recognition program basically tests the processing integrity of the system. This program requests that the subsystem found faulty or suspected of being faulty be diagnosed. The purpose of the diagnostic program is to generate test data to isolate the fault to a reasonably small section within the subsystem.

Real-time fault detection historically has been done at the circuit hardware level. In the early stages of development of fault-tolerant computers, attention was directed towards massive redundancy at the lowest level - the replication of individual components (resistors, transistors, etc.). The use of component redundancy has been limited by design difficulties and by new developments in componet technology. The change from discrete components to integrated circuits has largely invalidated the assumption of independent component failures. Without it,the advantages of component redundancy are lost.

The most developed techniques are fault detection by periodic diagnosis and the application of parity and similar error codes to detect or correct errors in data transmission and storage. The periodic diagnosis techniques have progressed from exclusively software implementations to software combinations with special-purpose hardware. Concurrent diagnosis uses error detecting codes and monitoring circuits.

A. HARDWARE

Fault detection and diagnosis by hardware have greatly increased the sensitivity and selectivity of finding and correcting errors. In the early days, fault detection systems utilized registers that read data at specific times into an output device.[1,2] Later specially built hardware devices set off alarms when an error was detected.[3] The circuit that produced the fault was located by utilizing a book which contained manual diagnosis. As designers progressed, circuits were designed to not only send interrupts to the computer notifying it of an error, but also allowing the diagnostic routine programmatic access of the error register.[4,5,6]

To assist in locating faults, the hardware system may be partitioned into logical subelements that allow reconfiguration. Accurate timing of events requires a real time clock in the system. Transient or permanent faults may be initially detected by hardware devices but efficient identification and location of the faults requires software diagnostic routines. Diagnostic routines are built upon the concept of detecting faults by executing one instruction at a time. As the instructions become more complicated, more circuitry (Microsteps) is analyzed for faults. This is repeated until all elements are insured to be fault free or a fault is located.

It can be seen that to effectively detect an error in a timely manner requires hardware circuits. Programmatic access to error registers allows greater flexibility and speed in diagnosing the actual fault.

B. SOFTWARE

Many studies and investigations have been made in the area of fault-detection and diagnosis by software. Even the earliest computer systems had diagnostic programs to check the computers for errors.[7] As computers became more complex, the size of diagnostic routines increased as did the time required to write them (in man years).[3,5,8,9] By combining fault-detection with diagnostic routines, the total time to locate an error was reduced. By allowing periodic maintenance checks to be performed using this combined method, large computer systems reduced their amount of down time.[10,11,12,13] By combining fault-detection and diagnostic routines with automatic system reorganization, the down time may be reduced to a minimum with imposed system degradation.[4,14,15]

Software must also be partitioned into logical subelements to allow for program relocation or reconfiguration. Knowing when to reconfigure requires monitoring the most critical data of each logical subelement program. When critical data of a program are detected as being faulty, then the program itself is faulty.

When the combined technique of fault-detection, diagnostic routines and automatic system reorganization is used in a large system, additional problems occur. Finding time to run the required tests is a problem in a heavily loaded system. If there is barely enough time to complete the required tasks, how can we allow extra time for maintenance tests? The answer implies some type of dynamic time allocation. Another problem is the manner of presenting this detailed data to the system monitor operator in a timely manner. The operator must have enough data, but in a short time, to allow him to complete the action required of him before the total system fails.

III. DEFINITIONS AND CONCEPTS OF FAULT MONITORING IN A MULTI-PROCESSING SYSTEM

Large and complex computer systems increase the demand for fault monitoring systems. Because of this complexity, man by himself requires too much time to solve the same problem. The cost of uncorrected errors is especially severe in large multi-computer systems and in situations which a computer controls a very valuable system, and is not readily accessible for human repair. Examples are a real-time control computer and a spacecraft computer controlling an inter-planetary mission. A second critical requirement for fault monitoring exists when human lives may be affected by computer errors, (e.g. military defense systems, high-speed transportation control systems, or medical systems).[14] The time to repair such complex system must be reduced, and fault monitoring is one approach.

The five necessary parts of fault monitoring are: (1) Partitioning, (2) Dynamic time allocation, (3) Fault recognition, (4) Diagnostic routines, (5) System reconfiguration. These must be considered in detail so that the total effect may result in an efficient optimal system monitor. The following description describes the necessary elements of a fault monitoring multi-computer system. Each part of fault monitoring will first be defined in detail. Secondly, some explicit uses of these parts will be given for fault monitoring.

A. PARTITIONING

Partitioning is the process by which a large complicated system is divided into logical subelements. Each subelement has a specific

function in either hardware or software. By requiring each subelement to be a logical subdivision, it may then be replaced in case of a detected failure.[4,10,16] An example is a program subroutine that is located in faulty computer memory core; it may be relocated to a fault free core location. Another example is a detected fault in an input/output (I/O) channel; the monitoring system would reconfigure the system input/outputs so as to utilize another channel. If a Central Processing Unit (CPU) failed, its tasks (processing of logical program subelements) could be allocated to other CPU's in a reduced operating mode.

The most important requirement for successful partitioning is to segment the system into logical elements, each having the capability of being relocated by the system reconfiguration module.[4,14,15] Thus it is required that core memories, for example, be divided into modules, regardless of the actual computer memory organization. The hardware items could be partitioned into CPU's, core memory modules and input/output channels. Thus when one hardware item fails and another similar hardware unit is free, (or only partly utilized) it may be used immediately by the system monitoring reconfiguration program. In a similar manner, all computer software programs and subprograms should be partitioned into logical units of approximately equal core size so that immediate reconfiguration may be performed. Programs of unequal size would create the problem of moving all programs in the computer. If a display device should fail and the display processor program becomes idle, the system could be reconfigured to use the teletype (or some other output device) along with the teletype processing module.

B. DYNAMIC TIME ALLOCATION

Time slicing is the division of the total computer system time available. This system time is allocated to all of its component parts and programs.[12,18] Time slicing is used to minimize the time required for fault detection and diagnosis in any one time frame. During routine system operations when the system is lightly loaded, there are large blocks of time available in each executive cycle for fault recognition and diagnostic analysis. As the system becomes more in demand, the time available for analysis is shortened. In order to utilize this time more effectively, the fault monitoring program must dynamically allocate the available time. Thus the program must know how much time is available for use and thus how much diagnosis may be performed in this specific cycle. Flags (or some other method) must be set and the proper bookkeeping performed to insure that the most critical data is still monitored and analyzed during the most heavily loaded period of system utilization. During lull periods, the monitoring program must also insure that all components of the computer system are analyzed so as to insure complete system integrity.[12,13]

It follows directly that fault detection routines must be timed and must be able to operate under flag (executive) control. By the proper allocation of these routines to the pertinent tasks at hand, all criteria may be satisfied. Strict timing control of the main and monitoring program must be performed and must be programmatically available. [13]

By dynamically controlling the time used for fault monitoring, a wide range of operational modes may be effectively monitored. These

modes may vary from lightly loaded systems to very heavily loaded systems. In a lightly loaded system most all operations are concentrated towards fault monitoring, diagnostic analysis and maintenance. Under heavy loading, the time for monitoring is very small. In most systems, it is zero except for hardware fault monitors. By dynamically allocating a small segment of time to special purpose fault monitoring routines, increases reliability may be gained with little system interference.

C. FAULT RECOGNITION

Fault detection in digital computers is implemented either by periodic or by concurrent diagnosis. The most common current approach is periodic diagnosis which utilizes a diagnostic program stored in the computer memory.[6,17] Computation is periodically interrupted and the diagnostic program is executed. The diagnostic program itself is vulnerable to faults in the memory system. The cost of diagnosis consists of: (1) the storage used for the diagnostic program, (2) the time consumed by its execution, (3) the time needed for repair, (4) the repeated execution (rollback) of the program segment which was run after the last diagnosis.[14] Such time and storage costs are very severe in real-time computing. The alternate diagnosis method is concurrent diagnosis in which error-detecting codes and monitoring circuits are employed to indicate the presence of faults.

A distinction must be made in fault detection between transient and permanent errors. By maintaining a history of detected errors with no diagnosable faults, a trend of transient errors may be stored. This trend may be utilized to determine an impending major fault. In

critical locations, special hardware devices must be installed to detect errors that would remain undetected or unrecoverable by software diagnosis. For example, the current instruction address, in the location counter, may be required to locate a fault. If the location counter is not programmatically available concurrent with the fault detection, then a special fault location register is required. This register would automatically copy the contents of the location counter at the time of any fault.[4,14]

Faults may originate from either hardware or software. They may also be detected by either hardware, software or a combination of both. A list of all the pertinent errors to be detected is required. From this list a division must be made between the two types of fault origin, hardware or software. This decision is influenced by the method of fault detection. This list is used in the process of determining partitions. Each fault detection technique used depends upon many system factors that must be taken into account. How the system is partitioned affects the grouping of faults. This grouping of faults is used by the dynamic time allocation routine.

D. DIAGNOSTIC ROUTINES

The diagnostic program is designed to isolate and specify errors in main-frame arithmetic and control logic, the various information transfers, the various devices and registers. The program is constructed on the general basis that every command in the machine repertoire uses a unique set of microinstructions or microsteps, leads to a correct result and a second command using the same set plus one leads to an incorrect result, then the failure is assumed to

be in the additional microstep. The detailed diagnosis of errors requires that the possiblilities of control signal failure, transmission path failure, and register failure, be investigated. It is often difficult to separate these types.[4,5,14,15]

For a large scale real-time multi-computer system, fault diagnosis routines become even more complex and time consuming than the diagnostic routine just described. To eliminate these problems, a modularized systems approach must be utilized. Rather than be concerned with a specific circuit element failing, we concentrate on detecting modular subsystem errors. Critical data are monitored so as to indicate failures in any one of our subsystem modules. Upon confirming a permanent error in a module, the system reconfiguration program is called.[4,14]

E. SYSTEM RECONFIGURATION

When permanent faults are detected and analyzed in a complex system, the total system may halt or the system may be reconfigured to avoid the faulting component (partitioned submodule) and operated at a reduced level.[4,14,15] As discussed before, halting a valuable system is not acceptable. Reconfiguration may be manual or automatic. In the automatic mode, the computer system must maintain a current configuration list of all submodules and their operational status. Upon notification of a submodule failing (that is part of the operational system), reconfiguration is forced. When a submodule is repaired, and proven operational, the system monitor operator may indicate this fact to the program and then request reconfiguration.[13] In the manual mode, the system must be examined manually, the new configuration determined and then manually implemented.

The reconfiguration program contains the necessary information to logically interconnect all submodules and to relocate computer programs when necessary. By displaying the proposed reconfigured computer system to the system monitor operator, approval may be given and the new system implemented. Automatic reconfiguration could save as much as 99 percent of the time required for manual reconfiguration.

While automatic reconfiguration gives an indication of overly complicating the problem, it is in reality a simplification. To manually maintain the configuration control of a large multi-computer system is a large team effort. Many charts, manuals, and switches must be coordinated with exacting precision. Automation of this task reduces this problem. The system configuration is maintained up to date in the computer memory. Switching and logical control of the input/output ports are controlled by computer subprograms. When these aids are implemented in a reconfiguration module program, the time required to change a configuration is reduced to seconds.

Because of the hypercritical nature of the fault monitoring process, special precautions must be observed. The fault monitoring program can be duplicated in another computer or reside in a special fault tolerant computer. These precautions reduce the danger of a fault occurring during execution of the reconfiguration program.

IV. A SPECIFIC EXAMPLE: A SHIP COMBAT WEAPON SYSTEM

One example of a complicated real-time multi-computer system is a ship combat weapon system. Many computers are utilized to solve special subsystem tasks, such as target detection and missile firing. The total computer complex is integrated together by the Combat Information Center System. In this real-time system, many control and data processing functions must be performed at extremely high speeds. Most of the sensory and control devices must be electrically connected on-line to the system to permit automatic transfer of data. Delays in transferring data by means of manual off-line handling of tapes, cards, etc., are not acceptable.

An executive control philosophy was developed for distributing the various tasks among the computers. Control of these tasks in each computer is maintained by an Executive Routine. Over-all control of the multi-computer system is not employed since each computer is controlled by its own Executive Routine. Subroutines or tasks are controlled by the Executive Routine by the use of flags or alerts. Decisions on whether to respond to the flag or alert at any specific time are determined by the priority of the input and the time available to do the task. This accurate timing is made possible by use of an internal real-time clock. When each task is completed, the flags and alerts are sensed again and the highest priority task remaining is performed. Both periodic and demand type tasks are utilized by executive routines.

A Ship Combat Weapon System is an example of a large real-time multi-computer system in actual use. A detailed study of this system will be explored for the methodology of Fault Monitoring techniques. The methods evolved for Fault Monitoring in a Ship Combat Weapon System will apply towards most large real-time multi-computer systems.

A. PROPERTIES OF A COMBAT WEAPON SYSTEM AND THE DATA INVOLVED

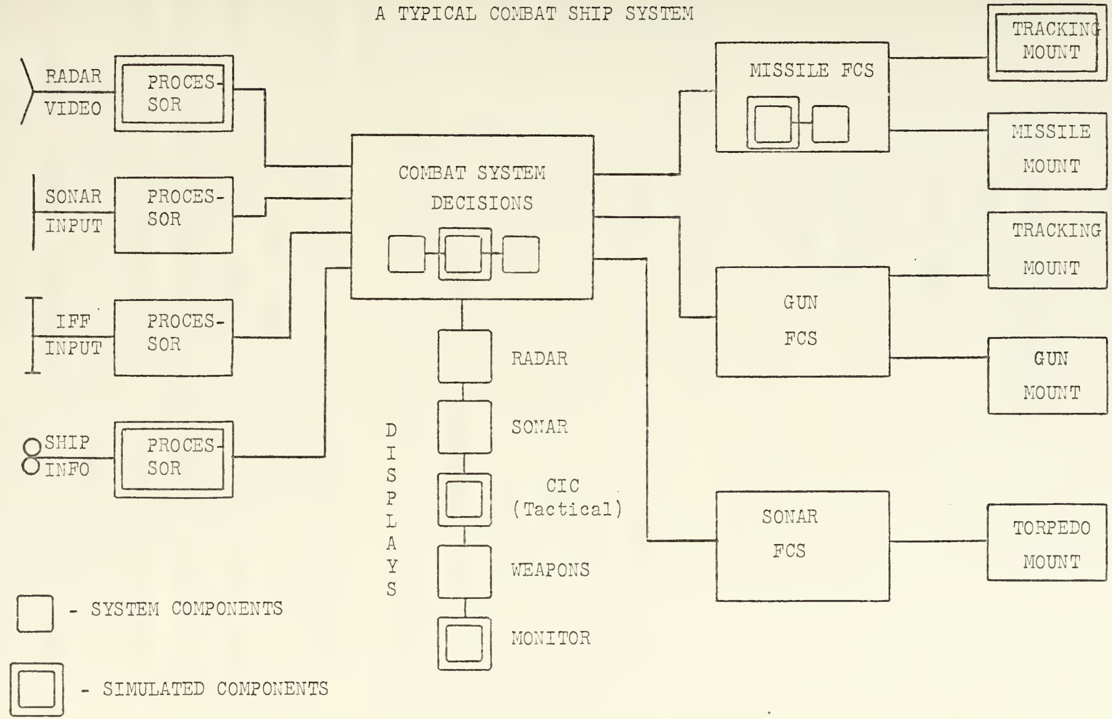
A typical Ship Combat System consists of many interconnected systems (see fig. 1). They are divided into three major areas: (1) Input, (2) Processing and (3) Output. All different types of input devices are analyzed by the input processors and the relevant data transmitted to the processing section. The processing section correlates the data from the different sensors. The correlated data is transferred to the appropriate output device (Guns, Missiles).

Each of these three areas have in turn many components, some of which are:

1. Inputs
    - A. Radar video
    - B. Sonar input
    - C. IFF input
    - D. Ship information
2. Processing (Combat System Decisions)
    - A. Three processing computers
    - B. Five types of display consoles
        - (1) Radar
        - (2) Sonar
        - (3) CIC (Tactical)

FIGURE 1

A TYPICAL COMBAT SHIP SYSTEM



(4) Weapons

(5) Monitor

3. Outputs

A. Missile Fire Control System

(1) Tracking mount

(2) Missile mount

B. Gun Fire Control System

(1) Tracking mount

(2) Gun mount

C. Sonar Fire Control System and Torpedo Mount

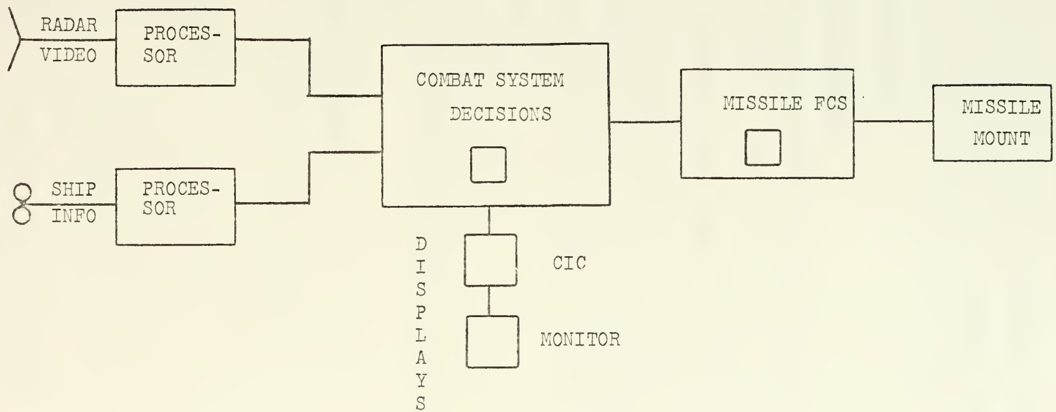
B. THE SIMULATED COMBAT WEAPON SYSTEM

The Ship Combat Weapon System described in figure 1 is very complicated. To analyze this system in detail for Fault Monitoring purposes is a large task. A representative system will be simulated instead (see fig. 2). The simulation will contain Input, Processing and Output sections. By including one system with each function, a representative but reasonably sized, heavily loaded system may still be simulated.

The Naval Postgraduate School has the unique computational facilities of a large system simulation laboratory with three digital computers and one analog computer: the Xerox Data Systems (XDS) 9300 medium scale digital computer, two Adage Graphic Terminals AGT-10, and the Comcor CI-5000 analog computer. Each digital computer is assigned a major task of the Ship Combat Weapon System while the analog computer (CI-5000) simulates the physical missile mount. The physical identity of the computers, the system functions and the computer programs may be identified in figure 3.

FIGURE 2

SIMULATION OF THE SHIP COMBAT SYSTEM



The Adage Graphic Terminal computer number one (Adage 1 computer) contains the Radar, the Ship processor and the Monitor programs. Adage Graphic Terminal computer number two (Adage 2 computer) contains the Combat System Decisions program. The XDS-9300 computer contains the Missile Fire Control System Program and the CI-5000 operates as the Missile Mount Simulation.

1. Types of Data Involved

Some typical types of data involved in a Ship Combat Weapon System are shown in table 1. Each of the five sections of table 1 are typical of the data that each system would contain. This data is utilized by the Fault Monitoring program to analyze the Combat System for the detection of errors. The fault monitoring program, will in turn pass this data to the Diagnostic program for further analysis and evaluation.

The following types of data may be sampled for error detection and fault location by an indication of large jumps in the data.

1. Radar azimuth
2. Target range and bearing
3. Gyro position, azimuth, pitch, and roll
4. Speed
5. Intercept point
6. Time to fire
7. Time to go
8. Launcher angle ordered
9. Launcher limits, data sample rate and bearing
10. Missile mount bearing and elevation

Normally, this data would be a continuous stream.

FIGURE 3 PARTITIONED HARDWARE

* - SIMULATED

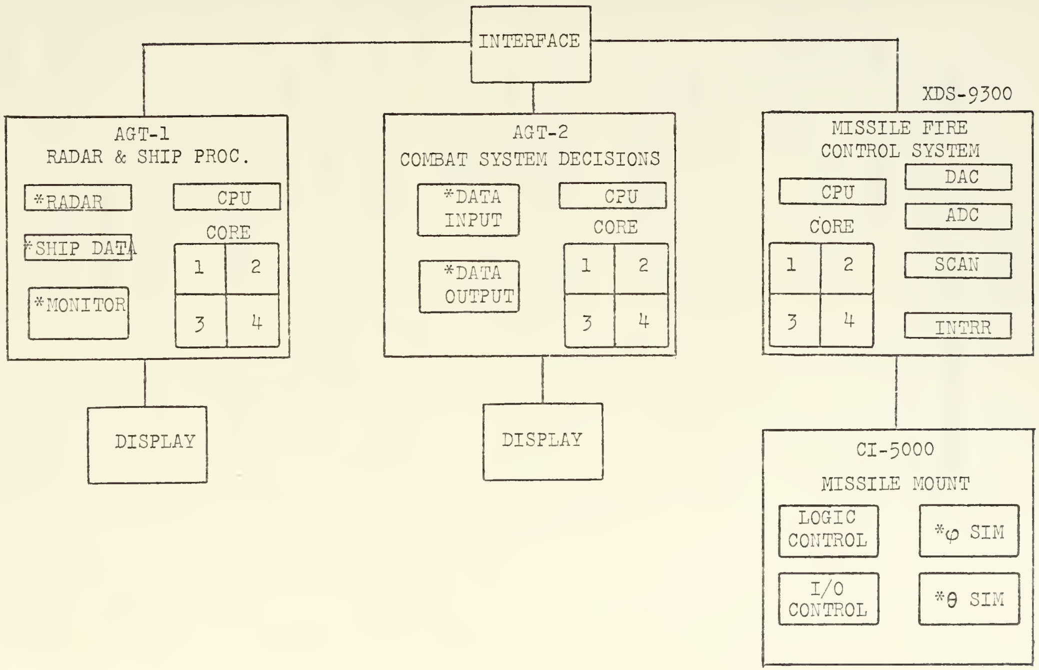


Table 1

Types of Data in the Simulated System

1. Radar Video & Processor
    A. Azimuth
    B. Target Data
        1. Range
        2. Bearing
2. Ship Information & Processor
    A. Gyro
        1. Position (Lat., Long.)
        2. Aximuth (Heading)
        3. Pitch, Roll
    B. Speed
3. Combat System Decisions
    A. Computer Status
        1. Memory Available
        2. I/O Channels available
        3. CPU's available
    B. Target Data (Speed, Heading)
    C. Total System Configuration
4. Missile Fire Control System
    A. Intercept Point
    B. Time to Fire
    C. Time to Go
    D. Target Destruction Evaluation

(Table 1 Cont.)

E. Launcher Angle Ordered

F. Launcher Data Sample Rate

G. Launcher Bearing Limits

5. Missile Mount

A. Bearing ($\theta, \dot{\theta}, \ddot{\theta}$)

B. Elevation ($\phi, \dot{\phi}, \ddot{\phi}$)

C. Operational Status

1. Errors in Bearing & Elevation

2. Drift Rates

Another method of error detection is by the analysis of data outside of some physical limits.

1. Angles greater than 360° or negative angles.

2. Target data outisde of radar range or negative range

3. Pitch greater than $\pm 20^{\circ}$

4. Roll greater than $\pm 90^{\circ}$

5. Ship speed greater than $\pm$ 100 knots

6. Acceleration greater than reasonable limits

A few general rules may be given to assist in the detection of faults.

1. Monitor data normally assumed to be continuous and smooth. When large deviations are detected, an error has occurred.

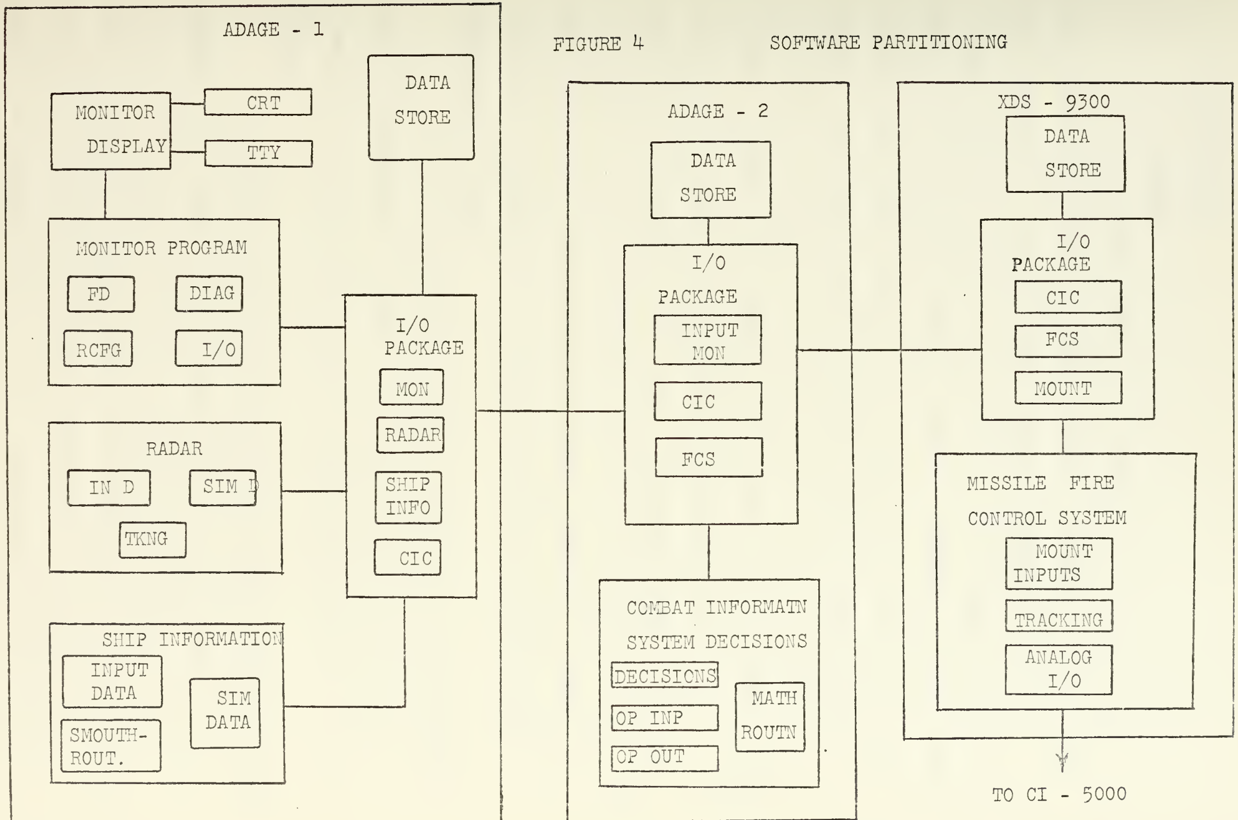
2. Monitor physical data and check for data outside physical limitations, i.e., ships moving faster than 100 knots.

3. Send test data to software subroutines and analyze the results.

4. Send test instructions to the computers to check for central processing errors.

The program interaction of the Ship Combat Weapon System is described in figure 4. Note the generalized use of the I/O package for data exchange between programs. This general use of a common program simplifies the automatic reconfiguration program. Since all programs have the same requirements of input and output, then data handling is the same. When data from one program looks the same to another program, then program relocation is simplified. A program may be moved from one computer to another without a change to the I/O package program. Data sampling and fault monitoring also become simpler for the same reason.

FIGURE 4 SOFTWARE PARTITIONING



2. Partitioning

a. Hardware

The hardware items of the simulated Ship Compat Weapon System were partitioned in section III into four major sections and many subsections. The four major computer subsystems have become the main hardware partitions. (see fig. 3) Each digital computer is composed of subsections normally ascribed to digital computers; Central Processor Units (CPU), magnetic core, display units and input/output channels. Some items like Radar, Monitor and input data have been simulated with software routines for lack of the actual hardware devices. Because of the similarity of program size and overall program action, partitioning by subsystems is a reasonable choice. By monitoring critical data within these subprograms, fault monitoring of a subsystem becomes simpler than monitoring the system in total. If any critical data from the radar subsystem is detected as erroneous, the direct assumption by the diagnostic routine is that the radar subsystem is at fault.

The analog computer contains a simulation of the Missile Mount and therefore, the actual gear train and motor systems are simulated. The logic control and I/O control are hardware accessable devices.

Note the similarity of the hardware items among the computer systems. This allows a more direct method for fault detection and analysis. Since all three digital computers have central processor units and all have modular computer core memories (or simulated ones), then they are similar for hardware partitioning. One module of a computer core memory could be used to replace another that contains

faults. The replacement could be in the same computer or in alternate computers. Automatic reconfiguration is possible with similar interchangeable subsystems.

Table 2 describes the critical data points in the Combat System used to detect the various hardware errors. With data monitoring of these hardware devices, any error may be analyzed to determine the actual device at fault.

b. Software

Programs have been partitioned into the same four major partitions as the hardware partitioning. Each subsystem function, such as Radar or Ship Information, is used to separate major partitions. (see fig. 4) The detailed partitions are different tasks within each subsystem function, such as Input data, Simulation data or Tracking. Because each partitioned programming task is of approximately the same magnitude, relocation is greatly simplified. Upon software program reconfiguration, the first step will be to reload the subprogram at fault into the same computer as the one it first faulted in. If this fails, the program will be reloaded into another computer with available space. Software reconfiguration will be completed faster this way than by reloading the total system.

The pertinent software errors for fault detection are shown in table 3. By diagnosing any of these errors, the faulting subprogram may be easily determined in a short time. The subprograms may then be relocated by the system reconfiguration program.

3. Interface Requirements

a. Data Paths

In order to examine the exchange of data in a detailed

Table 2

Hardware Errors to Detect

1. Radar

    A. Azimuth

    B. Target Data (range, bearing)

2. Gyro

    A. Heading

    B. Pitch, Roll

3. Pit Log

    A. Speed

4. Memory

    A. Read, Write

5. I/O Channels

    A. Parity

6. CPU

    A. Incorrect subroutine answers

7. Missile Mount

    A. Launcher Angle

    B. Bearing

    C. Elevation

    D. Drift rate

Table 3

Software Error Detection

1. Ship Information
   - A. Intercept Point
   - B. Time to Fire
   - C. Time to Go
2. Radar
   - A. Target Data - Speed, Heading
3. Gyro
   - A. Latitude
   - B. Longitude
   - C. Speed

manner, the actual electronic interface must be minutely analyzed. Table 4 describes the interface equipment that is involved in the system simulation program. Note the two different levels of core memory accessibility that are detailed for the XDS-9300 computer, (accessible and inaccessible). This is very appropriate since the modern modular computer system utilizes this technique of memory protection. Any data in this protected area of memory must first be accessed from inaccessable core and placed in accessable core.

Six levels of data accessability are described in table 4 to account for all possible types of interfaces. Some computer systems may only have one or two levels; the more complex systems may include all six types. Systems of all complexities will be represented by these levels of interfacing.

Table 5 lists the data paths and gives for each the access time, hardware path, interface interference and typical types of data that would be retrieved. The data paths are the same as shown in table 4. The access times are the actual times required for both the software and the hardware. The first time specified is a fixed time, the second is the time for each additional access. The column described as "hardware path" describes the actual path the data takes through the system. The right arrow (→) shows the path of the data from one hardware item to another as abreviated in table 4. The interface interference describes the interruption that the data access causes to the other hardware and software systems. The typical types of data retrieved are as described in table 1.

Table 4

INTERFACE EQUIPMENT COMPONENTS

		Abbreviation
1.	Adage memory (core)	AM
2.	Adage I/O program	AP
3.	Interface box (Adage to 9300)	IB
4.	9300 memory, accessible (8k-32k)	XMA
5.	9300 memory, inaccessible (0-8k)	XMI
6.	9300 I/O program	XP
7.	Hybrid interface box	HI
8.	Analog Computer (CI-5000)	AC

DEPTH OF DATA ACCESS (see table 5)
(Based on Monitor in Adage 1)

1.	Directly addressible	Adage memory (core)
2.	Indirectly addressible	Accessible core in 9300
3.	Programatically accessible (level 1)	Inaccessible core in 9300
4.	Programatically accessible (level 2)	Alternate Adage core
5.	Programatically accessible (level 3)	Analog data (DAC-ADC)
6.	Programatically accessible (level 4)	Analog data indirect (by SCAN system)

TABLE 5

DATA PATH	SOURCE	ACCESS TIME (u Sec)	HARDWARE PATH OF RETRIEVED DATA	INTERFACE INTERFERENCE	TYPICAL TYPES OF DATA THAT WOULD BE RETRIEVED (From Table 1)
1	ADAGE MEMORY	8.1 or +nx15.6	AM→AM	NONE	TARGET DATA (1B) GYRO, SPEED (2)
2	XDS ACCESSABLE MEMORY	370 +8(n-1)	AM→IB→XMA →IB→AM	INTERRUPTS XDS HARDWARE	INTERCEPT POINT (4A) LAUNCHER ANGLE ORDERED (4E)
3	XDS INACC. MEMORY	920 +25(n-1)	AM→IB→XMA →XP→XMI →XMA→IB→AM	INTERRUPTS XDS HARDWARE AND SOFTWARE	LAUNCHER DATA SAMPLE RATE (4F) LAUNCHER BEARING LIMITS (4G)
4	ALTERNATE ADAGE MEMORY	1.95msec +30(n-1)	AM_1→IB→XMA →XP→IB→AM_2 →IB→XP→XMA →IB→AM_1	INTERRUPTS XDS HARDWARE TWICE, ADAGE PROGRAM ONCE	TARGET DATA (3B) TOTAL SYSTEM CONFIGURATION (3C) (SEE FIGURE 5)
5	FIXED ANALOG DATA	1.0 msec +50(n-1)	AM→IB→XMA →XP→HI→AC →HI→XP→XMA →IB→AM	INTERRUPTS XDS PROGRAM TWICE	BEARING (5A) ELEVATION (5B)
6	RANDOM ANALOG DATA	18 m sec	AM→IB→XMA →XP→HI→AC(SP) →HI→XP→XMA →IB→AM	INTERRUPTS XDS PROGRAM TWICE AND ANALOG simulation once	ERRORS IN BEARING (5C1) DRIFT RATES (5C2)

b. A Specific Example

Figure 5 describes a specific example of retrieving data from a specific computer. Note the complex path that is necessary to access this data, ten data transfers in all. This is to be expected in large real-time systems and must be timed accurately. The data must first be requested from the data sampling program located in Adage 1. This request passes through the XDS-9300 computer to the Adage 2 computer. The Adage 2 computer must then access the requested data and pass it back to the Adage 1 computer via the XDS-9300 computer. While this data path is long, the majority of the time required is for program initiations that must be set-up (1.95 m sec). Thereafter, only a short time (30 μ sec) is required for each additional word retrieved, i.e., 100 words may be transferred in 3 m sec.

4. Monitor Program Module

The Monitor Program contains all items for Fault Monitoring as discussed. This program is divided into three related segments: (1) Program timing analysis and priority setup, (2) Data sampling and (3) Human operator interface. Each segment is independent, but relies upon completion of related tasks. Upon completion of all tasks, the cycle of monitoring the total system is complete.

a. Timing Analysis

This subprogram samples the system usage of time to allow the most efficient allocation among the required tasks. Any time that remains after the required tasks have been completed is surplus time. In most systems, this surplus time is not utilized. For example,

FIGURE 5

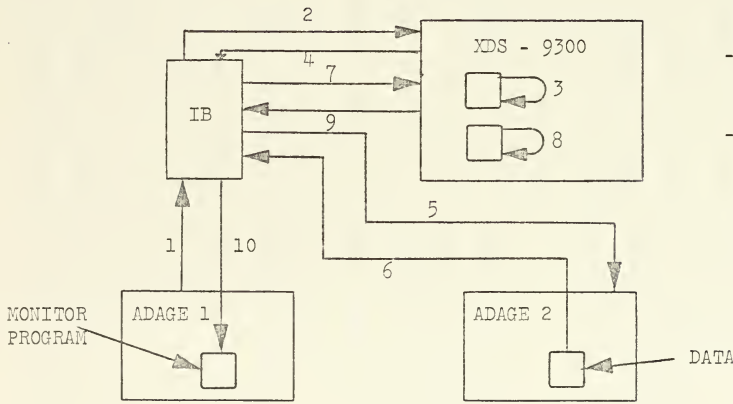
EXAMPLE OF DATA SAMPLING (Level 4)

1. To retrieve the data called "Total system configuration" (3C)
2. The data is required in the monitor program (located in Adage 1); it must be retrieved from the CIC data store (located in Adage 2).
3. The data flow commences as follows:

$$AM_1 \xrightarrow{1} IB \xrightarrow{2} XMA \xrightarrow{3} XP$$

$$\xrightarrow{4} IB \xrightarrow{5} AM_2 \xrightarrow{6} IB \xrightarrow{7} XP$$

$$\xrightarrow{8} XMA \xrightarrow{9} IB \xrightarrow{10} AM_1$$



TIME REQUIRED TO ACCESS, 1950 μ SECONDS.

if the timing analysis program detects three milliseconds of time left in a ten millisecond executive loop, it allocates as much of the three milliseconds to Fault Monitoring as is possible. The allocation of this time is used in three different modes.

(1) Light loading mode. When the computer system is only lightly loaded (say 30%), many fault monitoring tasks may be accomplished. With this amount of time available, many monitoring tasks normally accomplished under maintenance down time may be loaded by segments into the computer memory from the disc by the monitor program. Since the probability of this lightly loaded condition occurring for a reasonable amount of time is high, many time slots allotted to the fault monitor may be utilized in loading fault detection and diagnostic programs for later execution. Execution of these programs in addition to those discussed below maintains the computer system at its greatest reliability.

(2) Medium loading. When the computer system is at moderate loading (say 60%), little program swapping of monitor routines is allowed. All major system functions are monitored and routine maintenance tests are performed only periodically, a section at a time. By concentrating the monitoring function on detecting errors of major system functions, the up time reliability is greatly enhanced by insuring system operation. When an error is detected, a quick system reconfiguration reduces the Mean Time to Repair (MTTR) to near zero.

(3) Heavy loading. When a computer system is heavily loaded every millisecond is needed to maintain the system in operation. This is system utilization of about 90 percent. With such little time

available, most real-time multi-processor systems accomplish no software fault monitoring at all. The results is that the smallest error can cause the total system to fail. This is a very bad mistake! It is in this situation that on-line Fault Monitoring is needed the most. By monitoring only the most critical data points and completing this task over many time slices, an effective monitoring program can be carried out even under heavy loading. Timing analysis is most important when there is very little time available. The process of dynamic time allocation can be shown to be most effective for the heavily loaded system.[18] When a serious permanent fault occurs, more time may be utilized for diagnostic routines to accurately locate the fault. With the imminent prospect of system failure, the locating and correcting of the fault now has highest priority. Only very perishable data need be saved so that an operable system may be restored on system restart. Therefore, a computer system with adequate fault monitoring will have greatly enhanced system reliability even during critical periods of heavy loading.

b. Data Sampling

The data that is needed for fault monitoring is sampled through the data paths and stored efficiently in core or disc. Redundant data is filtered and only data permutations are actually stored. This process requires an intricate scheme for storing data since the core space and the time available are both critical. For example, the azimuth angular rotation of the missile mount is considered to be continuous. Rather than store all data points over several seconds (about 500 points), only one data point need be saved.

This would be the "old" azimuth angle and would be compared to the newly acquired angle. The difference would than be compared against a maximum allowed difference. Whenever this maximum difference was exceded, an error would be generated. Thus only three words ("old" angle, "new" angle, and maximum difference) must be stored compared to possibly 500.

All data samples needed for adequate fault monitoring are grouped into sections, only those subprograms required are brought into core and executed. For example, the data elements (table 1) needed to be monitored for fault detection and the related extraction times (from table 5) are shown below.

Data	Access Time
1. Radar azimuth	8.1 μ sec
2. Radar target range	8.1 μ sec
3. Radar target bearing	8.1 μ sec
4. Ship latitude	8.1 μ sec
5. Ship longitude	8.1 μ sec
6. Missile intercept point	370 μ sec
7. Missile time to fire	370 μ sec
8. Missile target destruction evaluation	370 μ sec
9. Missile mount bearing	370 μ sec
10. Missile mount elevation	370 μ sec

Each data point can be retrieved individually or as a group. The access time of an individual data item is different than that of a group. The best method to choose is the method that results in the smallest average access time per data element. For example, by

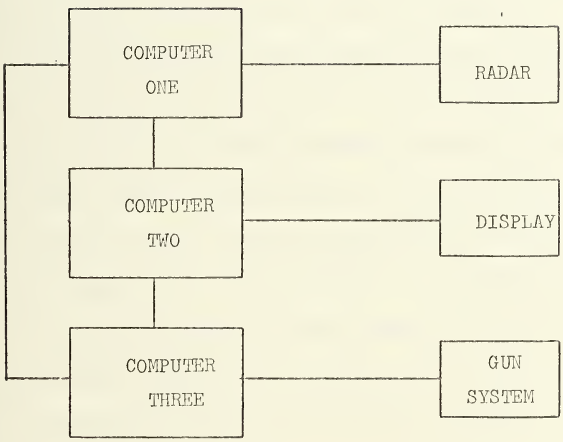
summing the individual access time for data items one through five, we obtain 40.5 μ seconds. Table 5 shows that if more than one data element of this type is accessed, the time of access is 15.6 μ seconds. Therefore by accessing data items one through five as a group, the required access time is 78.0 μ seconds. In this example, it is more advantageous to access each data item individually than by a group. Similarly summing the individual access time of items six through ten, we obtain 1,850 μ seconds. Again using table 5, we find that these same data elements accessed as a group require only 402 μ seconds. In this example, the data access time of a group is much less than that of the same items retrieved individually.

c. Human Operator Interface

The human operator interface module accepts the sampled data and analyzes the data for faults. Upon detecting a fault, the pertinent data is displayed to the operator as an alert. Requests from the operator are input into this module and displayed in the proper format. All human input actions, such as a light pen hit or function switch depressed are recognized by this program module and acted upon.

If reconfiguration is requested by the system monitor operator, a system study is conducted by the reconfiguration program module to analyze the current configuration. Then the program looks up the entry in the reconfiguration table appropriate to the component which has failed and displays the recommended reconfiguration for operator approval. (see fig. 6) If the operator approves this reconfiguration, he presses a function switch labeled "accept" and the program continues and executes the recommended reconfiguration.

PROPOSED RECONFIGURATION



SEMIAUTOMATIC MODE

IF ACCEPTABLE PRESS ACCEPT

IF NOT ACCEPTABLE MODIFY AS REQUIRED

FIGURE 6

If the system monitor operator disapproves of the recommended reconfiguration, he may alter the display by an appropriate manner and then order the computer to execute this new configuration.

5. Combat Information Center Program (SMULA)

A simulation of a Ship Combat Weapon System was programmed on the two Adage Graphic Terminals available at the Naval Postgraduate School. (see Program A and B) Three systems were simulated: (1) Combat Information Center, (2) Radar and (3) Ship Information.

The basic purpose of this simulation was to provide a model on which to test the ideas presented in the preceding section for a fault monitoring system. The simulation provides an actual model of combat between a ship and an aircraft. A display of the position of the ship and the airplane has been incorporated to allow visual following of the action. The simulation is programmed for the airplane to approach and attack the ship, firing missiles at the ship when close enough. The ship in turn must detect the airplane, and make the decisions of hostility, of time to fire and of target destruction.

a. Main Control and Combat Information Display (CICP)

This program operates in one Adage Graphic Display terminal (see Program A). The timing of the overall simulation is controlled in this module.

The command and control systems purpose is to accept data from the radar simulation and make a decision upon the identity of the target. If it is identified as hostile, a "kill" order is sent to the Fire Control Computer System Module. Since the time required for the decision process has a Poisson Distribution, it is simulated

by an constant eight second delay plus a random delay from an exponential random number generator with an expectation of four seconds.

As the fire control module rotates the missile launcher, the missile launcher displayed on the ship moves in synchronism with the missile launcher simulation on the Concor analog computer. After lock-on to the target, the combat system "launches" a salvo of two missiles. These are simulated on the Adage display as a pair of bright dots, one after the other, originating from the missile launcher and moving towards the target. If the missiles "hit" the target, the target explodes into many bits, simulated by many dots randomly spaced. If the plane launches a missile and "hits" the ship, the ship explodes in the same way. All missiles are simulated as nuclear type.

b. Radar and Ship Information Simulator (RADAR)

This program simulates the radar and ship information systems on the ship. It is located in one Adage Graphic Terminal. (see Program B)

The radar's purpose is to detect all incoming targets as soon as possible and relay information on range and bearing to the Combat System. For the simulated radar, a maximum range of one hundred miles was chosen. The delay between the time that the approaching aircraft corsses the point of maximum range and the time that the target data is actually transmitted, has a uniform random distribution with a maximum of eight seconds and a minimum of four seconds. Eight seconds was chosen after considering the antenna rotational speed and the number of rotations required for the radar operator to confirm an actual target.

The Ship Information subprogram simulates the ship as moving on a steady course at a speed of thirty knots. This information is passed to the CICP program and is used to move the simulated ship display.

The radar simulator has a model of a simulated airplane. The airplane model moves at a speed of 3,600 knots. The airplane starts on a course of 270° T. When it is within 100 miles of the ship, it turns to automatically attack the ship. Manual override controls are provided to control the course, altitude and missile firing. These are controlled by function switches adjacent to the display.

6. <u>Fire Control System Program (Missile Mount)</u>

A simulation of a Digital Fire Control System and of a Missile launching Mount was programmed on the Xerox Data Systems (XDS) 9300 digital computer and the Comcor CI-5000 analog computer. The Digital Fire Control System was simulated on the XDS-9300 computer (see Program C). The Missile Launching Mount was simulated on the CI-5000 analog computer (see Program D).

The basic purpose of these simulations was to provide a Digital Fire Control System and a Missile Mount to interact with the simulated Ship Combat Weapon System on the Adage Graphic Terminals. The Digital Fire Control System was written in FORTRAN IV on the XDS-9300 computer and uses its hybrid capabilities to communicate with the CI-5000 analog computer. The Missile Mount is simulated to act like a real mechanical-electrical missile mount. Upon assignment of a target azimuth, the missile mount moves as a missile mount aboard ship would move.

a. Digital Fire Control System

The digital fire control system accepts target information from the Combat System and converts this rectangular coordinate data to polar coordinates for the fire control missile mount. It must then order the missile mount to move from its present azimuth to the target azimuth. In a total analog system, this would be all that would be required; the analog feedback system would effect the required movement. In a digital system many improvements may be gained. Overshoot and time to rotate can be minimized under digital control. The digital control utilizes a modified "Bang-Bang" approach that uses six phases. Each phase implements a separate portion of the task of moving the missile mount. With appropriate programming, the missile mount moves at the fastest speed possible with the smallest overshoot. Digital Control optimizes the control of this simulated large and massive Missile Mount as it does in the real case.

b. Missile Launcher Mount

The missile mount was simulated on the CI-5000 analog computer using hybrid computer techniques. The simulated mount consists of an amplidyne controlled generator that drives a large motor which in turn drives a 100:1 gear train connected to the missile mount. The amplidyne requires 33 volts per field ampere and in turn controls the field coil of the generator that can produce 25 amperes at 440 volts. The generator drives a 200 horsepower motor at speeds up to 1150 RPM. The weight of the mount is 28 tons and may rotate at a rate of up to one radian per second. The resulting transfer and analog computer equations of the fourth order system are:

I $\ddot{e}_{gf} = -112\ \dot{e}_{gf} - 3030\ e_{gf} + 3.35\ (10^4)\ e_{af}$

II $\dot{I}_{gf} = -.3333\ I_{gf} + .0333\ E_{gf}$

III $I_a = 87\ I_{gf} - 7.67\ \dot{\theta}_m$

IV $\ddot{\theta}_m = .032\ I_a - .0909\ \dot{\theta}_m$

V $E_{af} = K\ (\theta_r - \theta_l)$

Where g - generator m - motor

l - launcher f - field

a - armature K - constant

Both azimuth and elevation controls are implemented on the analog computer and have been verified to be similar to a shipboard missile mount. The analog simulation adds realism to the combat system and allows actual hardware items to be monitored by the fault monitoring system.

V. RECOMMENDED TECHNIQUES

By analyzing the process used in fault monitoring in the simulated Ship Combat Weapon System, the overall technique should now be clear. By applying the following techniques, a Real-Time Multi-Computer Monitoring system may be designed to operate effectively even under heavy loading conditions.

A. DYNAMIC TIME ALLOCATION

The hardware and software data transfer rate between all hardware components must be accurately determined by an interface timing study. This may be accomplished by writing a simple program loop passing data between the components. All critical data (critical to the hardware and software partitions) must be determined and listed as either hardware or software accessable data needed for data evaluation. From the interface timing study, the time required to access this data may be determined. From this data list, groups of data should be determined so as to best fit the minimum time allotted to fault detection under heavily loaded conditions. This grouping must be done in conjunction with the study of Partitioning. When the final list is completed and all groupings made, this data becomes the basis of the data sampling program module.

The timing analysis program module works directly with the data sampling program. By analyzing the system resources, time allocation may be distributed to a number of data sampling group subprograms and data analysis programs. For example, two milliseconds may be allocated to monitor all critical data points of the radar system.

B. PARTITIONING

The multi-computer system must be partitioned into hardware and software logical subelements. By determining the degree of reconfiguration possible, e.g., the number of CPU's, the degree of partitioning becomes known in part because partitioning and reconfiguration are interrelated. Both must be determined together in order to optimize system resources. If two partitioned elements may not be used interchangeably for reconfiguration, then the partitioning is too small. Partitioning must also consider what data inside a proposed subelement is critical. Normally each logical element of a multi-computer system has a number of data elements that can be used to determine when the logical element has failed. These data elements are the critical data points of this partitioned logical subelement.

If a computer program is critical to the operation of a multi-computer system and has no replacement, then a simulation of the program should be included in the system. A simulation of a program may be a smaller version of the replaced program or it may be a dummy program that allows the total system to remain operational at a reduced level. Then a software fault in this program, that can not be corrected by the relocation of the program, may be corrected temporarily by the use of the simulation. While the simulation is maintaining the system at a degradated level, the error in the program may be corrected and the program then reinstated into the system.

After the system has been satisfactorily partitioned, the subsystem elements become the system status list. The logical connections (I/O) of these partitions are then also fixed and inserted into tables in the reconfiguration program module.

C. FAULT RECOGNITION

Since the critical data has been determined under the study of Dynamic Time Allocation, only the method of fault recognition remains. Software errors are only recognized by software routines, but hardware faults are best detected by a combination of hardware and software. If hardware devices are present to detect the faults, then they should be used in preference to software routines as hardware detection is much faster. If some faults require an exorbitant amount of time to be recognized in software, then the use of special purpose hardware registers and fault detectors should be studied. Special purpose hardware fault detectors operate at a much higher speed but may be more expensive than software subroutines.

Permanent faults and errors may be detected and analyzed by hardware or software, but transient faults and errors may only be economically analyzed by software routines. Provisions for detecting and analyzing transient errors must be included in the system.

D. DIAGNOSTIC ROUTINES

Diagnostic routines become smaller in fault monitoring programs that recognize faults at the subsystem level. Normally a fault may be due to any one of hundreds of likely components. All components must be diagnosed to determine which component is at fault. Since any subsystem may only have from three to five critical data points (for example), the diagnostic routine necessary to locate a subsystem error may be simpler in design.

Normal diagnostic routines, used to diagnose computers and special hardware devices, are utilized in this fault monitoring system also.

Since most of these routines require run times of minutes, they must be segmented into logical time elements that may be called by the system dynamic time allocation routine and executed when the computer is lightly loaded. In this way complete diagnostic analysis of the total multi-computer system can be accomplished.

E. SYSTEM RECONFIGURATION AND PRESENTATION

Items A-D above compile all the necessary data for system reconfiguration and presentation to the System Monitor Operator. The system status list that was generated from the study of Partitioning may now be used to determine when and how a system might be reconfigured. A set of possible configuration lists or even a program that computes an acceptable reconfiguration is available for the use of the reconfiguration program. When requested, the program studies the submodule that has failed to see if it is on the current subsystem active list. (see fig. 7 for a sample list) If it is, it removes it and places it on the non-active list. The program searches the possible configuration lists until it finds a match with the current subsystem active list. If a match is not found, it notifies the system monitor operator and halts. Before continuing, if the semi-automatic mode is set, the reconfiguration is presented to the operator for approval. In the automatic mode, this step is omitted. By resetting the logical interface list, any faulty input/output ports are bypassed. If needed, a program may be relocated (by reloading it). If the fault is programmatic, a suitable simulation program may be loaded to replace the faulty one.

The type of presentation displayed to the monitor operator depends upon the equipment available and the system being monitored. Since

EXAMPLE OF A SUBSYSTEM ACTIVE LIST

ACTIVE	COMPONENT (see fig. 3)
yes	RADAR
yes	SHIP DATA
yes	MONITOR
yes	CPU #1
yes	CORE 1a
yes	CORE 1b
yes	CORE 1c
no	CORE 1d
yes	DISPLAY 1
yes	INTERFACE 1
yes	INPUT DATA
yes	OUTPUT DATA
yes	CPU #2
yes	CORE 2a
yes	CORE 2b
no	CORE 2c
no	CORE 2d
no	DISPLAY 2
yes	INTERFACE 2
yes	DAC
yes	ADC
.	.
.	.
etc.	etc.

Figure 7

the data to be presented is voluminous and time critical, any mecahnical device would be too slow. Some type of Cathode Ray Tube (CRT) with function switches and maybe a typewriter input is needed. Then the fault, its location and recommended solutions can be displayed simultaniously. The recommended reconfiguration presentation can be either displayed as a logic diagram showing the reconfigured components and their links (see fig. 6) or the two lists (old and new reconfiguration lists) can be displayed side by side. Because of the rapid visual assimilation by the operator of the displayed data, rapid decisions may be made.

```
                         COMPUTER PROGRAM A

          MAIN CONTROL AND COMBAT INFORMATION DISPLAY

                              (CICP)


**********************************************************************

 C      THIS PROGRAM SIMULATES A REAL TIME SYSTEM,
 C      A DLGN-36, USS CALIFORNIA, AND A F-111 FIGHTER.
 C      THE SIMULATION MODELS A HIGH SPEED ATTACK ON THE
 C      SHIP, THE PLANE DETECTS THE SHIP AT 40 MILES
 C      AND RELEASES AN ANTI SHIP MISSILE AT 20 MILES.
 C      THE SHIP DETECTS THE PLANE AT 100 MILES AND
 C      RELEASES 2 ADVANCED TERRIER MISSILES AS SOON
 C      AS POSSIBLE WITH THE DETERMINED DELAYS.
 C      ALL SYSTEMS ARE SIMULATED WITH THE DELAYS
 C      BEING CALCULATED BY THE COMPUTER.
 C
 C    H.D.K.  8/12/69
 C    D.A.T.  UBC  10/20/69
 C    E. A. POTTER   1/9/70   MOD 70-11-9

**********************************************************************
```

```
 EXPUNGE
TITLE CICP
ENTRY CICP,THRST,BØX,RANDM,CLØCK,RØTATE,INITL,STARSGEN
ENTRY INI1,LØØP,SUN,1SHIP,1SH4,1SH41,1SH2,1SH6,2SHIP
ENTRY 2SH4,2SH41,2SH2,2SH6,CØLL,TRYSTARS3,TRP1,TRP5,MSTAR
ENTRY STARMØVE,SHIP1,SHIP1X,AM,FIRECØNTL,STØVER,DISPT,ØUTPUT,DISPDATA
ENTRY EXPØF,DATASTØRE,CTHETA,THETA
ENTRY GETIB,PUTJB
MACRØ1 SCL; SCLXX ENDM
MACRØ1 LØØP; LØØPX ENDM
MACRØ1 SWITCHØFF (NN,LØC)     [TØ TEST SWITCH ACTIØNS
MDAR'N SWITH
17500JH 13.+NN                [ARLS. + NN
NØØP
JPAN LØC
ENDM
MACRØ1 DISPLAY (SCALE,DX,DY,DZ,LØC)
IFSAME (LØC)()                [IF NULL, DØ NØTHING
ELSE
MDAR'N DISDN
JPAN .-1                      [DISPLAY DØNE
ARMD DISDN
MDAR'L
MD05 LØC                      [START AVG WITH LIST (LØC)
ARMD EØVPV
ENDC
MDAR SCALE
15420JH PS                    [MULT LØWER
NØØP
MDAR'L'A
77777JH 0
71100JH                       [AR11'F
26000JH DZ                    [MD06 (SET INTENSITY)
MDAR DX
15420JH PS                    [MULT LØWER
NØØP
```

```
MDAR'L'A
77777√H
ARMD DAT
MDAR DY
15420√H PS
NØØP
17440√H 15.     [ARRS 15.
NØØP
MDAR'L'A
77777
MDAR'Ø DAT
67100√H
MD10'A AVGØF  [STØP CLØCK
MD10'Ø AVGØN  [START AVG
MDIR EØVPV                        [MDDS LØC
ENDM
MD05=25000√H; MD07=27000√H; MD10=30000√H
SPL1=1; SPR1=5; SPH1=9.; SPF1=13.; SPT1=0
SPL2=16.; SPR2=12.; SPH2=8.; SPF2=4; SPT2=-1
 [CLKØN = CLØCK SETTING,  ARØPV = ØVERFLØW PIVØT
CLKØN=1000√H; FRMPV=77755; EØVPV=77756; EØLPV=77757; ARØPV=77771
THRST: 14                         [THRUST
TVC: 4                            [
HYPRA: 120.                       [HYPER TIME A
TTIME: 300.                       [TØRPEDØ TIME
PØFS: 1000                        [PØINT ØF FIRE START
TVFL: 110                         [TØRPEDØ VELØCITY
HYPRB: 120.                       [HYPER TIME B
EXP: 60.                          [TIME FØR EXPLØSIØN
GØNE: 200.                        [TIME TØ BE GØNE
TPCT: 32.                         [TØRPEDØ CØUNT
SUNR: 600                         [SUN RADIUS
SHPCR: 25                         [SHIP CIRCLE FØR CØLISIØN
TSHCR: 770                        [TØRPEDØ TØ SHIP CIRCLE RADIUS
TSHC2: 500
```

```
TTCR: 500                         [TØRPEDØ TØ TØRPEDØ RADIUS
SCL: 2000                         [SCALE FACTØR FØR SHIPS
PS: 23000                         [PØSITIVE FULL BRIGHTNESS
YAW: 40                           [YAW RATE
NSTRS: -1                         [STARS ØN FLAG
STRXM: 10
SUNBM: 0
RLT: 200                          [TIME ØF RELEASE ØF TØRPS
FUEL: 2400.                       [AMØUNT ØF FUEL
TEXP: 6000                        [SIZE FØR ESPLØSIØN
KGRAV: 100                        [AMØUNT ØF GRAVITY
HYRT: 120.
DAT: 0
IREPEAT Q,(1,2)
RLT\Q: 0
GØNE\Q: 0
EXP\Q: 0
HYPA\Q: 0
HYPB\Q: 0
XACC\Q: 0
YACC\Q: 0
FUEL\Q: -0
TVC\Q: -0
XVEL\Q: 0                         [TWØ LISTS 1 AND 2
YVEL\Q: 0
XGRV\Q: 0
YGRV\Q: 0
HYRT\Q: 0
TPCT\Q: 0
SHF\Q: 0
SH\Q\CR: 0
Q\PTBF: LØC .+100.                [1 PØINT BUFFERS
RNDL\Q: REPEAT 8., 0; ENDR [8 RANDØM NUMBERS EACH
ENDI
BØX: 07000/H 54000
```

```
07000JH 54000
37776JH 54001
37776JH 40001
07000JH 40001
07000JH 54001
07001JH 54001
TØRPS: REPEAT 91., 1JH; ENDR
TØRPA: LØC .+140                 [STØRAGE FØR TØRPEDØES
YAW1: 20000
YAW2: 60000
XPØS1: 60000  [STARTING DATA FØR FIGURES
YPØS1: 60000
XPØS2: 57777
YPØS2: 57777
INIT: 0
DØNEF: 0
STRTF: 0                         [INIT -0
SUNB: -0
DISDN: -0
STPTR: STARS-1                   [STAR PØINTER
FTA: 0                           [INIT TØRPA +134
TLA: 0
TEM1: 0                          [INIT TØRPA +4
TEM2: 0
TPTR: 0
ARØVF: 0                         [AR ØVERFLØW STØRAGE
SSCL: 0
PTR1: 0
PTR2: 0                          [TØRPEDØ PØINTERS
PTR3: 0
PTR4: 0
PTR6: 0
PTR7: 0
PTR8: 0
LPTR: 0
```

```
6.1     SPTR: 0                          [INIT TØRPS -1
6.2     PREV: 0
6.3     SWITH: 0                         [SWITCH H
6.4     SWITL: 0                         [SWITCH L
6.5     NSTR: 0                          [NØ STARS
6.6     SDFLG: 0
6.7     BRCNT: 0
6.10    STITEM: 0
6.11    BØXF: 0                          [BØX FLAG
6.12    GRVF: 0                          [GRAVITY FLAG
6.13    STRF: 0
6.14    KILSN: 0
6.15    ZERØ:C0: 0
6.16    AMASK: 77777
6.17    ØNE:C01: 1
6.20    EØLBIT: 1√H 0
6.21    CM11: -(1√H 1)
6.22    SEVEN: 7
6.23    C11: 1√H 1
6.24    MZRØ: -0
6.25    HMZRØ: 77777√H
6.26    GHAØF: -20√H
6.27    FS: 37777                        [DISPLAY SCALE
6.30    HFS: 20000
6.31    AVGØF: CLKØN  [01000
6.32    AVGØN: 60740√H
6.33    RANDM: JUMP . [SIMPLE RANDØM NUMBER GENERATØR
6.34    MDAR'H RNDN
6.35    MDAE'F'B 53110
6.36    MDXØ .-1
6.37    ARMD RNDN
6.40    MDIR RANDM
6.41    RNDN: 0
6.42    RNDM2: JUMP . [THREE STEP RANDØM NØ GEN
6.43    JPSR RANDM
```

```
17500√H 5
MDAR'A'I RNDM2
ARMD RNDT
JPSR RANDM
JPSR RANDM
17500√H 7
MDAR'A'I RNDM2
MDAE RNDT
MDAE'I'N RNDM2
MDIR'X RNDM2
RNDT: 0
ARØSB: JUMP . [FØR ØVERFLØW
ARMD'Ø ARØVF
MDIR ARØSB
NULL: JUMP .  [FØR INDERRECT JUMP
JUMP'I NULL
CLØCK: JUMP . [UPDATES CLØCK
ARMD EØLAR
MDAR'H'X DØNEF
JPAN .+2
ARMD'Ø STRTF
MDAR EØLAR
JUMP'I CLØCK
EØLAR: 0
EØL: JUMP .
ARMD'Ø DISDN
JUMP'I EØL
CGRAV: JUMP . [CALCULATES GRAVITY
MDIR CGRAV
RØTATE: JUMP .              [CALCULATES RØTATIØN
ARMD XYRØTATE
MDAR'A'L
77777
ARMD YRØTATE
MDAR'H XYRØTATE
```

```
10.1      MDAR'L'A
10.2      77777
10.3      ARMD XRØTATE
10.4      15420/H CØSN
10.5      NØØP
10.6      ARMD TEMRØTATE
10.7      MDAR'N SINE
10.10     15420/H YRØTATE
10.11     NØØP
10.12     MDAE TEMRØTATE
10.13     MDAR'A'L
10.14     77777/H 0
10.15     ARMD TEMRØTATE
10.16     MDAR YRØTATE
10.17     15420/H CØSN
10.20     NØØP
10.21     ARMD YRØTATE
10.22     MDAR XRØTATE
10.23     15420/H SINE
10.24     NØØP
10.25     MDAE YRØTATE
10.26     17440/H 15.
10.27     NØØP
10.30     MDAR'L'A
10.31     77777
10.32     MDAR'Ø TEMRØTATE
10.33     MDAR'A'L
10.34     77776/H 77776
10.35     ARMD TEMRØTATE
10.36     MDAR'L
10.37     1/H 1
10.40     MDAR'A XYRØTATE
10.41     MDAR'Ø TEMRØTATE
10.42     MDIR RØTATE
10.43     TEMRØTATE: 0
```

```
XRØTATE: 0
YRØTATE: 0
XYRØTATE: 0
CICP: JUMP .  [ENTRY PØINT
ARXØ'F
ARMD'N STRF                     [TØ DELETE STARS AND GRAVITY
ARMD'N GRVF
MDAR'F DATASTØRE
ARMD DATASTØRE                  [END ØF ADDITIØN
MDAR'I CICP
JPLS .+2
JUMP INITL-4
ARMD THRST
MDAR'I'X CICP
ARMD TVEL
MDAR'X'I CICP
ARMD KGRAV
MDAR'L; 57777 [SET INITIAL PØSITIØN ØF SHIP
ARMD XPØS2
ARMD YPØS2
INITL: MDAR'F NULL              [START ØF INITALIZATIØN
ARMD FRMPV                      [STARTS CLØCK
MD10 ZERØ
JPSR STØVER                     [TRANSFERS INIT IF SET
MDAR'F EØL
ARMD EØLPV
MDAR'L
JPSR ARØSB
ARMD ARØPV
MDAR'F STARS
ARMD'L
STARSPTR: 0
STARSGEN: JPSR RANDM
MDAR'A CM11
ARMD'X'I STARSPTR
```

```
12.1        ARMD'X'I STARSPTR
12.2        MDAR'Ø ØNE
12.3        ARMD'X'I STARSPTR
12.4        MDAR'F STARSEND
12.5        MDAE'N STARSPTR
12.6        JPAN .+2
12.7        JUMP STARSGEN
12.10       MDAR STARS2-1
12.11       MDAR'Ø EØLBIT
12.12       ARMD STARS2-1
12.13       MDAR STARS3-1
12.14       MDAR'Ø EØLBIT [ADDS EØL BIT
12.15       ARMD STARS3-1
12.16       MDAR STARSEND-1
12.17       MDAR'Ø EØLBIT
12.20       ARMD STARSEND-1
12.21       ARXØ'F                        [SETS CØRE TØ ZERØ
12.22       ARMD RLT1
12.23       ARMD GØNE1
12.24       ARMD EXP1
12.25       ARMD HYPA1
12.26       ARMD HYPB1
12.27       ARMD XVEL1
12.30       ARMD YVEL1
12.31       ARMD XGRV1
12.32       ARMD YGRV1
12.33       ARMD HYRT1
12.34       ARMD RNDL1
12.35       ARMD RNDL1+1
12.36       ARMD RNDL1+2
12.37       ARMD RNDL1+3
12.40       ARMD RNDL1+4
12.41       ARMD RNDL1+5
12.42       ARMD RNDL1+6
12.43       ARMD RNDL1+7
```

```
13.1        ARMD'Ø TVC1
13.2        ARMD RLT2
13.3        ARMD GØNE2
13.4        ARMD EXP2
13.5        ARMD HYPA2
13.6        ARMD HYPB2
13.7        ARMD XVEL2
13.10       ARMD YVEL2
13.11       ARMD XGRV2
13.12       ARMD YGRV2
13.13       ARMD HYRT2
13.14       ARMD RNDL2
13.15       ARMD RNDL2+1
13.16       ARMD RNDL2+2
13.17       ARMD RNDL2+3
13.20       ARMD RNDL2+4
13.21       ARMD RNDL2+5
13.22       ARMD RNDL2+6
13.23       ARMD RNDL2+7
13.24       ARMD TCØUNT
13.25       ARMD THET2F
13.26       ARMD TØRP2F
13.27       ARMD FCS2F
13.30       ARMD SIMF2
13.31       ARMD SIMF3
13.32       ARMD SIMF4
13.33       ARMD SIMF5
13.34       ARMD SIMF6
13.35       ARMD SIMF7
13.36       ARMD SIMF8
13.37       ARMD SIMF9
13.40       ARMD SIMF10
13.41       ARMD SIMF11
13.42       ARMD SIMF12
13.43       ARMD SIMF13
```

```
14.1        ARMD SIMF14
14.2        ARMD SIMF15
14.3        ARMD SIMF16
14.4        ARMD SIMF17
14.5        ARMD SIMF18
14.6        ARMD DATATEMP
14.7        ARMD'Ø TVC2
14.10       ARMD INIT
14.11       ARMD IBLK                  [SET TØ ZERØ FØR AGT2
14.12       ARMD TLA
14.13       ARMD TØRPA
14.14       ARMD STRTF
14.15       ARMD'Ø DISDN
14.16       ARMD'Ø SUNB
14.17       MDAR'L
14.20       20000
14.21       ARMD YAW2
14.22       MDAR'L; 40000
14.23       ARMD YAW1
14.24       MDAR'L; 37777
14.25       ARMD XPØS1
14.26       MDAR'L; 77777
14.27       ARMD YPØS1
14.30       MDAR'L; -20
14.31       ARMD XVEL1
14.32       MDAR'L; 1
14.33       ARMD YVEL2
14.34       ARMD SIMF1                 [ADDED
14.35       MDAR'L; 3000
14.36       ARMD THETA
14.37       MDAR SIMTEN
14.40       ARMD'N SIMTIME
14.41       MDAR'N TPCT
14.42       ARMD TPCT1
14.43       ARMD TPCT2
```

```
15.1        MDAR'N FUEL
15.2        ARMD FUEL1
15.3        ARMD FUEL2
15.4        MDAR'L
15.5        TØRPA+4
15.6        ARMD TEM1
15.7        INI1: MDAR TEM1              [INITIAL 1
15.10       MDAS'F'N 4
15.11       ARMD'I TEM1
15.12       MDAR TEM1
15.13       MDAS'F 4
15.14       ARMD TEM1
15.15       MDXØ'F TØRPA+140
15.16       JPLS INI1
15.17       MDAR'L
15.20       TØRPA+134
15.21       ARMD FTA
15.22       MDAR'F TLA
15.23       ARMD LPTR
15.24       MDAR'F CLØCK
15.25       ARMD FRMPV
15.26
15.27       MDAR'H'X NAMEC
15.30       JPAN STN4
15.31       MDAR'L; -400
15.32       ARMD DØNEF
15.33       MDAR .-2
15.34       ARMD NAMEC
15.35       STN2: JPSR CØFF
15.36       JPSR SUP
15.37       STNAM
15.40       STNAM
15.41       NAME
15.42       MDAR'H'X NAMEC
15.43       JPAN STN2
```

```
JPSR C0N
JUMP .
STNAM: N00P
JUMP'I STN1
STN1: .+1
MDAR'H'X D0NEF
JPAN STN2-2
JPSR C0FF
MDAR'L
MTEN: -10.
ARMD NAMEC

STN4: MD10'0'L
CLK0N

JUMP STN3
NAMEC:-0
NAME:
N0CARRET
ASCII '△N△N△N△N△E△E△C△F△X△060△Y△070△' CICP  △'
△C△F△F△X△050△S△N A C0MBAT WEAP0N △C△F△N
△X△050△N△N SYSTEM SIMULATI0N  △C△F△F△N
△X△064△N△N△S WRITTEN BY △C△E△F
△X△074△N△N LT E. A. P0TTER △C△F△S
△X△100△N△N JUNE 1970, NPGS ';
CARRET
STN3: ARMD'0 D0NEF              [SET UP C0UNTER
JPSR RANDM                      [GET RAND0M NUMBERS
MDAR'N STRTF  [WAIT UNTILL FRAME INTERRUPT
JPAN .-2
L00P: MDAR'F 77775              [0VERALL L00P 0F PR0GRAM
ARMD D0NEF                      [SETS UP C0UNTER
MDAR INIT
ARMD'H IBLK                     [SET F0R AGT-2
JPAN INITL
```

```
17.1     ARX8'F
17.2     ARMD STRTF
17.3     MDAR SWITH
17.4     ARMD SWITL
17.5     MDIC'8'L
17.6     40√H
17.7     S5AR'F'H              [READS BUTT8N INT8 SWITH
17.10    17500√H 1
17.11    MDAR'L'A
17.12    1√H 77776
17.13    ARMD SWITH            [PUTS IT INT8 SWITH
17.14    MDIC'A'L
17.15    -40√H
17.16    S5MD TEM1
17.17    MDAR'H TEM1
17.20    MDAR'A 8NE
17.21    MDAR'8 SWITH
17.22    ARMD SWITH
17.23    MDAR TEM1
17.24    MDAR'A'H'L
17.25    6
17.26    MDAR'8 SWITH
17.27    ARMD SWITH
17.30    MDAR SWITL
17.31    MDX8'A SWITH
17.32    ARMD TEM2
17.33    MDAR'H'N TEM2
17.34    17500√H 4
17.35    JPAN .+3              [TESTS F8R B8X DISPLAY
17.36    MDAR B8XF
17.37    ARMD'N B8XF
17.40    MDAR'H'N TEM2
17.41    17500√H 8.
17.42    JPAN .+3              [TESTS F8R STARS DISPLAY
17.43    MDAR STRF
```

```
20.1     ARMD'N STRF
20.2     MDAR'H SWITH
20.3     ARMD'H IBLK                  [FOR AGT-2
20.4     JPAN INITL                   [TESTS FOR NEW GAME
20.5     17500√H 1
20.6     NOOP
20.7     JPAN .+2
20.10    JUMP DOBOX
20.11    JUMP DOBOX
20.12    DISPT: JUMP .
20.13    ARMD NUMBER
20.14    MDAR'N'L; 4
20.15    ARMD BCOUNTER [DISPLAYS TCOUNT
20.16    MDAR ONETWOF
20.17    JPLS .+3
20.20    MDAR'F NUMBER1-1
20.21    JUMP .+2
20.22    MDAR'F NUMBER2-1
20.23    ARMD STOLAC                  [INITILIZE TO NUMBER -1
20.24    MDAR NUMBER
20.25    ARMD SAVEN
20.26    TESTC: MDAR SAVEN
20.27    17500√H 3                    [SHIFT 3 LEFT FOR 1 ST NUMBER
20.30    ARMD SAVEN
20.31    ARAR'H'F
20.32    JPLS INSERTN  [ALSO TEST FOR LEADING BLANKS
20.33    MDAR BLANK
20.34    INSERT: ARMD'I'X STOLAC
20.35    MDAR'N'X'H BCOUNTER          [TEST FOR END OF NUMBERS
20.36    JPAN DISEND                  [DONE
20.37    JUMP TESTC                   [SHIFT 1 LEFT FOR NUMBER
20.40    INSERTN: MDAR'A NMASK        [GET ONLY NUMBER
20.41    MDAR'B SIXTY  [ADD IN 60
20.42    ARAR'B'F                     [SHIFT 1 LEFT FOR CORRECT POSIT
20.43    JUMP INSERT
```

```
DISEND: MDAR STØLAC
MDAS'F'N 1
ARMD STØLAC
MDAR'I STØLAC
MDAR'Ø DECPT
ARMD'I STØLAC [ADD IN DEC PT
MDIR DISPT
STØLAC: 0
BCØUNTER: 0
BLANK: 100
SIXTY: 60
NMASK: 7
DECPT: 134/H
SAVEN: 0
NUMBER: 0
ØNETWØF: 0
DISPDATA: JUMP .
JPSR FIRECØNTL
JUMP .
DISPO: 0
MDAR'N CTHTF  [TEST FØR REFIGURED DISPLAY
JPAN DPEND
ARMD CTHTF
ARXØ'F
ARMD ØNETWØF
MDAR MILES
17500/H 2
JPSR DISPT
MDAR'F 1
ARMD ØNETWØF
MDAR TCØUNT
17500/H 2
ARMD TEMPDISP [STØRE TEMP IN TEMPDISP
MDAS'N DATATEMP
MDAS'H TEMPDISP
```

```
NØØP                                    [REMØVED ARMD'I'X DATASTØRE
MDAR TEMPDISP
ARMD DATATEMP [STØRE NEW IN ØLD AND DISPLAY
JPSR DISPT
MDAR'N EXP1
MDAR'I'H'Ø DATASTØRE                    [ØR IN EXP1 AND RESTØRE
NØØP                                    [REMØVED ARMD'I DATASTØRE
DPEND: JPSR FC2
JUMP .
DISP9: NØØP
MDIR DISPDATA [RETURN
DISP8: NØØP
JUMP'I DISP7
DISP7: DISP9
DISP1: DISP0
DISP2: NØØP
JUMP'I DISP1
MILES: 65432  [SET UP INIT FØR TEST
TEMPDISP: 0
DATATEMP: 0
MACRØ1 DATADISP (FL,ADD,DADR)
MDAR FL
JPLS .+2
JUMP ADD
MDAR'H SIMTIME
JPAN .+10
ARXØ'F
ARMD FL
MDAR FLGØNE
ARMD FL+1
JPSR EXPØF
ARMD'N SIMTIME
JUMP ADD
MDAR'F DADR
JPSR DISPDATA
```

```
ENDM
MACRØ1 UPDATE (FL,ADD)
MDAR FL
JPLS .+2
JUMP ADD
MDAR'H SIMTIME
JPAN ADD
ARXØ'F
ARMD FL
MDAR FLGØNE
ARMD FL+1
MDAR'N DISPTIME
ARMD SIMTIME
JPSR CTHETA
ARXØ'F
ARMD'N CTHTF
ENDM
STØVER: JUMP .
ARAR'X SIMTIME
ARAR'X SIMTIME2
ARAR'X SIMTIME3
MDAR SIMF1
JPLS .+2
JUMP DISWD1
MDAR'H SIMTIME
JPAN DISWD1
MDAR'N SIMTEN
ARMD SIMTIME
ARMD SIMTIME3
JPSR CTHETA
MDAR FLGØNE
ARMD SIMF19
MDAR'N'H MILES
MDAE'L; 2700/H
JPAN  DISWD1
```

```
24•1     ARX8'F
24•2     ARMD TC8UNT
24•3     ARMD SIMF1
24•4     MDAR FLG8NE
24•5     ARMD SIMF2
24•6     ARMD SIMF4
24•7     ARMD THET2F
24•10    JPSR RNDM2
24•11    77
24•12    MDAS'L; 100
24•13    ARMD'N SIMTIME
24•14    MDAR YAW1
24•15    MDAS'N THETA
24•16    ARMD'H THET2
24•17    JUMP END8VER
24•20    DISWD1: MDAR SIMF2
24•21    JPLS •+2
24•22    JUMP DISWD2
24•23    MDAR YTANF
24•24    JPLS •+5
24•25    MDAR'H YAW1
24•26    MDAE THET2
24•27    JPAN DISWD2
24•30    JUMP •+4
24•31    MDAR'H'N YAW1
24•32    MDAE'N THET2
24•33    JPAN DISWD2
24•34    ARX8'F
24•35    ARMD THET2F
24•36    ARMD SIMF2
24•37    MDAR FLG8NE
24•40    ARMD SIMF3
24•41    DISWD2: MDAR SIMF3
24•42    JPLS •+2
24•43    JUMP DISWD3
```

```
MDAR'N EXP2
JPAN DISWD3
ARX0'F
ARMD SIMF3
JPSR RESTF
JUMP DISWD15
DISWD3: UPDATE (SIMF4,DISWD4)
MDAR'N EXP1
JPAN .+2
JPSR RESTF
MDAR'F WD2
JPSR DISPDATA
DISWD4: DATADISP (SIMF5,DISWD5,WD2)        [DISPLAY TARGET DETECTED
DISWD5: UPDATE (SIMF6,DISWD6)
MDAR'N EXP1
JPAN .+2
JPSR RESTF
MDAR'F WD3
JPSR DISPDATA
DISWD6: DATADISP (SIMF7,DISWD7,WD3)        [DISPLAY DICISION TO KILL
DISWD7: UPDATE (SIMF8,DISWD8)
MDAR'N EXP1
JPAN .+2
JPSR RESTF
MDAR'F WD4
JPSR DISPDATA
DISWD8: MDAR SIMF9
JPLS .+2
JUMP DISWD9
ARX0'F
ARMD CTHTF
JPSR CTHETA
MDAR YAW2
MDAS'N YAWC2
ARMD YAW2
```

```
MDAR'H YAW2
MDAE'H'N THETA
JPAN .+4
MDAR'F WD4
JPSR DISPDATA [DISPLAY TRACKING
JUMP DISWD9
ARXØ'F
ARMD SIMF9
ARMD SIMTIME
MDAR FLGØNE
ARMD SIMF10
JUMP ENDØVER
DISWD9: UPDATE (SIMF10,DISWD10)
MDAR'N EXP1
JPAN .+2
JPSR RESTF
MDAR'F WD5
JPSR DISPDATA
DISWD10: DATADISP (SIMF11,DISWD11,WD5) [DISPLAY LØCKØN
DISWD11: UPDATE (SIMF12,DISWD12)
MDAR'N EXP1
JPAN .+3
JPSR RESTF
JUMP DISWD12
MDAR FLGØNE
ARMD FCS2F
ARMD SIMF15
MDAR'F 27                        [TIME TØ WAIT
ARMD'N SIMTIME2                  [UNTIL MISSILES FIRED
MDAR THETA
ARMD YAW2
MDAR'F WD6
JPSR DISPDATA
DISWD12: MDAR SIMF13
JPLS .+2
```

```
27.1     JUMP DISWD13
27.2     MDAR'H SIMTIME2
27.3     JPAN DISWD13
27.4     ARX0'F
27.5     ARMD FCS2F
27.6     ARMD SIMF13
27.7     JPSR CTHETA
27.10    MDAR MILES
27.11    17440JH 2
27.12    ARMD'N SIMTIME2
27.13    MDAR FLG0NE
27.14    ARMD SIMF14
27.15    DISWD13: MDAR SIMF14
27.16    JPLS .+2
27.17    JUMP DISWD14
27.20    MDAR'N EXP1
27.21    JPAN .+10
27.22    ARX0'F
27.23    ARMD FCS2F
27.24    ARMD T0RP2F
27.25    ARMD SIMF3
27.26    ARMD SIMTIME
27.27    JPSR RESTF
27.30    JUMP DISWD15
27.31    MDAR'H SIMTIME2
27.32    JPAN DISWD14
27.33    ARX0'F
27.34    ARMD SIMF14
27.35    ARMD SIMF15
27.36    MDAR FLG0NE
27.37    ARMD SIMF9
27.40    JUMP DISWD8
27.41    DISWD14: DATADISP (SIMF15,DISREP,WD6)
27.42    DISREP: ARX0'F
27.43    ARMD SIMF16
```

```
DISWD15: UPDATE (SIMF16,DISWD16)
ARMD FCS2F
ARMD TØRP2F
MDAR'F WD7
JPSR DISPDATA
DISWD16: DATADISP (SIMF17,DISWD17,WD7) [DISPLAY DESTRØYED TARGET
DISWD17: MDAR SIMF18
JPLS .+2
JUMP DISWD18
ARXØ'F
ARMD SIMF18
ARMD SIMF2
ARMD SIMF3
DISWD18: MDAR SIMF19
JPLS .+2
JUMP ENDØVER

JPSR $RØWFW                    [WRITE ØUT IBLK
        -0
       115
    IBLK
GETIB: 10.

JPSR $RØWFW                    [READ IN JBLK
         0
       116
     JBLK
PUTJB: 10.
JPSR $FINSH

ENDØVER: MDIR STØVER
RESTF: JUMP . [SUBPRØGRAM TØ RESET
ARXØ'F                        [AND TØ DISPLAY
ARMD SIMF4                    [DISPLAY TARGET DISTRØYED
ARMD SIMF5
```

```
31.1      ARMD SIMF6
31.2      ARMD SIMF7
31.3      ARMD SIMF8
31.4      ARMD SIMF9
31.5      ARMD SIMF10
31.6      ARMD SIMF11
31.7      ARMD SIMF12
31.10     ARMD SIMF13
31.11     ARMD SIMF14
31.12     ARMD SIMF15
31.13     MDAR FLG9NE
31.14     ARMD SIMF16
31.15     MDIR RESTF
31.16     SIMF1: 0
31.17     SIMF2: 0
31.20     SIMF3: 0
31.21     SIMF4: 0
31.22     SIMF5: 0
31.23     SIMF6: 0
31.24     SIMF7: 0
31.25     SIMF8: 0
31.26     SIMF9: 0
31.27     SIMF10: 0
31.30     SIMF11: 0
31.31     SIMF12: 0
31.32     SIMF13: 0
31.33     SIMF14: 0
31.34     SIMF15: 0
31.35     SIMF16: 0
31.36     SIMF17: 0
31.37     SIMF18: 0
31.40     SIMF19: 0
31.41     FCS2F: 0
31.42     T8RP2F: 0
31.43     THET2F: 0
```

```
THET2: 0
FLGØNE: 1
SIMTEN: 10
SIMTIME: 0
SIMTIME2: 0
SIMTIME3: 0
DISPTIME: 100
FIRECØNTL: JUMP .
ARMD CHRSTRING
JPSR CØFF
JPSR SUP
DISP2
DISP2
CHRSTRING: WD2
JPSR CØN
MDIR FIRECØNTL
FC2: JUMP .
ARMD FIRESAVE
MD07'L; 0
JPSR CØFF
JPSR SUP
DISP8
DISP8
NUMBER1-3
JPSR CØN
MDAR FIRESAVE
MDIR FC2
CØFF: JUMP .
MDIC'A'L
-10
MD10'A'L
776JH
MDIR CØFF
CØN: JUMP .
MD10'Ø'L
```

```
33•1         776√H
33•2         MDIC'8'L
33•3         10
33•4         MDIR C8N
33•5         SUP: JUMP •
33•6         MDAR'I SUP
33•7         ARMD 77736
33•10        MDAR'X SUP
33•11        MDAR'I SUP
33•12        ARMD 77737
33•13        MDAR'X SUP
33•14        MDAR'I SUP
33•15        ARMD 77735
33•16        MDAR'X SUP
33•17        MDIR SUP
33•20        FIRESAVE: 0
33•21        MD06= 26000√H
33•22        AR07= 27100√H
33•23        MD11= 31000√H
33•24        WD2: 0                       [TARGET DETECTED
33•25        0463005670
33•26        1743000000
33•27        5220251216
33•30        4265020210
33•31        4265042606
33•32        5221342100
33•33        WD3: 0                       [DECISI9N T8 KILL
33•34        0463005670
33•35        1743000000
33•36        4221241622
33•37        5162247634
33•40        2025047500
33•41        4562346230
33•42        WD4: 0                       [TRACKING
33•43        0463005670
```

```
34.1        1743000000
34.2        5224440606
34.3        4562347216
34.4        WD5: 0
34.5        0463005670
34.6        1743000000
34.7        2023047606              [LØCK ØN
34.10       4550147634
34.11       WD6: 0                  [FIRE MISSILES
34.12       0463005670
34.13       1743000000
34.14       4322251212
34.15       2023244646
34.16       5162246212
34.17       0024700000
34.20       WD7: 0                  [DESTRØYED TARGET
34.21       0463005670
34.22       1743000000
34.23       4221251650
34.24       5123654612
34.25       4210052202
34.26       5121742650
34.27       0
34.30       0464005700              [NEW X,Y VALUES
34.31       1742000000              [UP BRIGHTNESS,ITALICS
34.32       NUMBER1: 0
34.33       0
34.34       0
34.35       0
34.36       0
34.37       2023244630              [ MILES, TIME = .
34.40       4264600000
34.41       0463005710              [NEW X,Y VALUES
34.42       5222246612
34.43       2017220000
```

```
NUMBER2: 0
0
0
0
0
2024642606                      [ SEC
0000100000   [END ØF DATA AND TEST
DØBØX: MDAR BØXF
JPAN SUN
DISPLAY (FS,CO,CO,HFS,BØX)
SUN: NØØP
1SHIP: MDAR'H ØNE
ARMD SHTV1
ARXØ'F
ARMD XACC1
ARMD YACC1
MDAR SCL
ARMD SSCL
MDAR RLT1
JPLS .+2
JUMP .+2
ARAR'X RLT1
MDAR GØNE1
JPLS 1SH5
MDAR EXP1
JPLS 1SH4
MDAR HYPA1
JPLS 1SH6
MDAR HYPB1
JPLS 1SH7
SWITCHØFF (SPR1,TURNR1) [THIS IS THE TURN RØUTINE
MDAR YAW1
MDAS'N YAW
ARMD YAW1
TURNR4: JPSR SNCØS
```

```
MDAR'L;PTEN: 20
15420√H CØSN
NØØP
ARAR'H
ARMD XVEL1
MDAR PTEN
15420√H SINE
NØØP
ARAR'H
ARMD YVEL1
JUMP TURNR3
TURNR1: MDAR THET2F
JPLS .+2
JUMP .+4
MDAR YTANF                    [TEST FØR NEG DIRRECTIØN
JPLS TURNR4-3
JUMP TURNR5
SWITCHØFF (SPL1,TURNR2)
TURNR5: MDAR YAW1
MDAS YAW
ARMD YAW1
JUMP TURNR4
TURNR2: MDAR YAW1
JPSR SNCØS
TURNR3: SWITCHØFF (SPT1,1NEXT)
1NEXT: MDAR TVC               [DISPLAYS FIRE
ARMD'N TVC1
JPSR RANDM
MDAR'Ø'L
77777√H 60000
MDAS'F'N 17777
MDAR'Ø C11
ARMD'H SHTV1
1SH4: MDAR JBLK               [PLANE
ARMD XPØS1
```

```
37.1      MDAR JBLK+1
37.2      ARMD YPØS1                    [ADDED FØR READ FRØM ADG-2
37.3      MDAR JBLK+2
37.4      ARMD YAW1
37.5      MDAR JBLK+10
37.6      ARMD TØRP2F
37.7      MDAR HYRT1
37.10     JPLS .+2
37.11     JUMP .+2
37.12     ARAR'X HYRT1
37.13     MDAR EXP1
37.14     JPLS 1SH3
37.15     1SH10: NØØP
37.16     1SH40: NØØP
37.17     1SH42: NØØP
37.20     1SH41: NØØP
37.21     1SH11: SWITCHØFF (SPH1,1SH2)
37.22     MDAR HYRT1
37.23     JPLS 1SH2
37.24     MDAR HYPRA
37.25     ARMD'N HYPA1
37.26     JUMP 1SH99
37.27     1SH2: MDAR TØRP2F
37.30     JPLS 1SH2P
37.31     SWITCHØFF (SPF1,1SH20)
37.32     1SH2P: MDAR TPCT1
37.33     MDAR TPCT1
37.34     JPLS .+2
37.35     JUMP 1SH20
37.36     MDAR RLT1
37.37     JPLS 1SH20
37.40     MDAR FTA
37.41     JPLS .+2
37.42     JUMP 1SH20
37.43     ARMD TEM1
```

```
40.1        ARAR'X TPCT1
40.2        MDAR'I TEM1
40.3        ARMD FTA
40.4        ARXØ'F
40.5        ARMD'I TEM1
40.6        MDAR TEM1
40.7        ARMD'I LPTR
40.10       ARMD LPTR
40.11       MDAR'L; 140
40.12       ARMD'I'X'N TEM1
40.13       MDAR PØFS
40.14       15420JH SINE
40.15       NØØP
40.16       MDAE'H YPØS1
40.17       MDAR'A'H AMASK
40.20       ARMD'H TEM2
40.21       MDAR PØFS
40.22       15420JH CØSN
40.23       NØØP
40.24       MDAE'H XPØS1
40.25       MDAR'A'H AMASK
40.26       MDAR'Ø TEM2
40.27       ARMD'I'X TEM1
40.30       MDAR TVEL
40.31       15420JH SINE
40.32       NØØP
40.33       MDAE'H YVEL1
40.34       MDAR'A'H AMASK
40.35       ARMD'H TEM2
40.36       MDAR TVEL
40.37       15420JH CØSN
40.40       NØØP
40.41       MDAE'H XVEL1
40.42       MDAR'A'H AMASK
40.43       MDAR'Ø TEM2
```

```
ARMD'I'X TEM1
MDAR RLT
ARMD'N RLT1
1SH20: MDAR'L
SHIP1X-1
ARMD'L
SHIP1XPTR: 0
MDAR'L
SHIP1-1
ARMD'L
SHIP1PTR: 0
SHIP1R8T: MDAR'I'X'H SHIP1XPTR
JPLS .+2
JUMP 1SH20A
ARAR'H
JPSR R8TATE
ARMD'I'X SHIP1PTR
JUMP SHIP1R8T
1SH20A: MDAR'L
MD05 SHIP1
1SH21: ARMD TEM1
MDAR'N DISDN
JPAN .-1
ARMD DISDN
MDAR TEM1
ARMD E8VPV
DISPLAY (SSCL,XP8S1,YP8S1,HFS)
JUMP 1SH99
1SH5: MDAR'X G8NE1                [TEST F8R EXPL8SI8N
JPLS 1SH99
ARMD'8 INIT
JUMP 1SH99
1SH6: MDAR'X HYPA1
JPLS 1SH99
MDAR HYPRB
```

```
42•1      ARMD'N HYPB1
42•2      JPSR RANDM
42•3      MDAR'A AMASK
42•4      ARMD YPØS1
42•5      JPSR RANDM
42•6      MDAR'A AMASK
42•7      ARMD XPØS1
42•10     ARXØ'F
42•11     ARMD'H 1PTBF
42•12     ARMD'H 1PTBF+1
42•13     MDAR'Ø'H ØNE
42•14     ARMD'H 1PTBF+2
42•15     ARMD'H 1PTBF+3
42•16     MDAS'F 1
42•17     ARMD'H 1PTBF+4
42•20     JUMP 1SH30
42•21     1SH7: MDAR'X HYPB1
42•22     JPLS 1SH30
42•23     JPSR RANDM
42•24     17500√H 7
42•25     MDAR'A SEVEN
42•26     MDAE'L
42•27     MDAR RNDL1
42•30     ARMD •+1
42•31     NØØP
42•32     JPAN 1SH14
42•33     ARMD'I'Ø •-2
42•34     MDAR HYRT
42•35     ARMD'N HYRT1
42•36     JUMP 1SH99
42•37     1SH30: MDAR'L
42•40     MD05 1PTBF
42•41     JUMP 1SH21
42•42     1SH14: MDAR EXP
42•43     ARMD'N EXP1
```

```
ARX0'F
ARMD XGRV1
ARMD YGRV1
JUMP 1SH99
1SH3: MDAR'X EXP1
JPLS .+4
MDAR 0NE
ARMD'N G0NE1
JUMP 1SH99
QQ=0
REPEAT 7,
JPSR RNDM2
17777/H 17777
MDAR'A CM11
ARMD 1PTBF+QQ
ARMD 1PTBF+1+QQ
MDAR'0 0NE
ARMD 1PTBF+2+QQ
QQ=QQ+3
ENDR
MDAR'H 0NE
ARMD 1PTBF+QQ
MDAR EXP1
MDAS'F 120.
15440/H 6
N00P
ARMD SSCL
JUMP 1SH30
SSCL2: 10000
YAWC2: 100
1SH99:
2SHIP: MDAR'H 0NE                [SHIP
ARMD SHTV2
ARX0'F
ARMD XACC2
```

```
ARMD YACC2
MDAR RLT2
JPLS .+2
JUMP .+2
ARAR'X RLT2
MDAR GØNE2
JPLS 2SH5
MDAR EXP2
JPLS 2SH4
MDAR HYPA2
JPLS 2SH6
MDAR HYPB2
JPLS 2SH7
SWITCHØFF (SPR2,.+4)
MDAR YAW2
MDAS'N YAWC2
ARMD YAW2
SWITCHØFF (SPL2,.+4)
MDAR YAW2
MDAS YAWC2
ARMD YAW2
MDAR YAW2
JPSR SNCØS
JUMP 2SH4
2SH4: NØØP
MDAR JBLK+3
ARMD XPØS2
MDAR JBLK+4                    [ADDED FØR READ FRØM AGT-2
ARMD YPØS2
MDAR HYRT2
JPLS .+2
JUMP .+2
ARAR'X HYRT2
MDAR EXP2
JPLS 2SH3
2SH10: NØØP
```

```
2SH40: N0OP
2SH42: N0OP
2SH41: N0OP
2SH11: SWITCH0FF (SPH2,2SH2)
JPSR CTHETA                               [BY HYPER BUTT0N
2SH2: MDAR FCS2F
JPLS 2SH2P
SWITCH0FF (SPF2,2SH20)
2SH2P: MDAR TPCT2
JPLS .+2
JUMP 2SH20
MDAR RLT2
JPLS 2SH20
MDAR FTA
JPLS .+2
JUMP 2SH20
ARMD TEM1
ARAR'X TPCT2
MDAR'I TEM1
ARMD FTA
ARX0'F
ARMD'I TEM1
MDAR TEM1
ARMD'I LPTR
ARMD LPTR
MDAR TTIME
ARMD'I'X'N TEM1
MDAR P0FS
17500√H 1
15420√H SINE
N0OP
MDAE'H YP0S2
MDAR'A'H AMASK
ARMD'H TEM2
MDAR P0FS
```

```
15420JH CØSN
NØØP
MDAE'H XPØS2
MDAR'A'H AMASK
MDAR'Ø TEM2
ARMD'I'X TEM1
MDAR TVEL
15420JH SINE
NØØP
MDAE'H YVEL2
MDAR'A'H AMASK
ARMD'H TEM2
MDAR TVEL
15420JH CØSN
NØØP
MDAE'H XVEL2
MDAR'A'H AMASK
MDAR'Ø TEM2
ARMD'I'X TEM1
MDAR'F 14
ARMD'N RLT2
2SH20: MDAR'F SHIP2X-1
ARMD'L
SHIP2XPTR: 0
MDAR'F SHIP2-1
ARMD'L
SHIP2PTR: 0
SHIP2RØT: MDAR'I'X'H SHIP2XPTR
JPLS .+2
JUMP 2SH20A
ARAR'H
JPSR RØTATE
ARMD'I'X SHIP2PTR
JUMP SHIP2RØT
2SH20A: MDAR'L
```

```
MDO5 SHIP2
2SH21: ARMD TEM1
MDAR'N DISDN
JPAN .-1
ARMD DISDN
MDAR TEM1
ARMD EØVPV
DISPLAY (SSCL2,XPØS2,YPØS2,HFS)
JUMP 2SH99
2SH5: MDAR'X GØNE2              [CHECK FØR FØR EXPLØSIØN
JPLS 2SH99
ARMD'Ø INIT
JUMP 2SH99
2SH6: JUMP 2SH99
2SH7: JUMP 2SH99
2SH30: MDAR'L
MDO5 2PTBF
JUMP 2SH21
2SH14: MDAR EXP
ARMD'N EXP2
ARXØ'F
ARMD XGRV2
ARMD YGRV2
JUMP 2SH99
2SH3: MDAR'X EXP2
JPLS .+4
MDAR GØNE
ARMD'N GØNE2
JUMP 2SH99
QQ=0
REPEAT 7,
JPSR RNDM2
17777JH 17777
MDAR'A CM11
ARMD 2PTBF+QQ
```

```
50.1    ARMD 2PTBF+1+QQ
50.2    MDAR'8 8NE
50.3    ARMD 2PT3F+2+QQ
50.4    QQ=QQ+3
50.5    ENDR
50.6    MDAR'H 8NE
50.7    ARMD 2PTBF+QQ
50.10   MDAR EXP2
50.11   MDAS'F 120.
50.12   15440JH 6
50.13   N8OP
50.14   ARMD SSCL
50.15   JUMP 2SH30
50.16   2SH99:
50.17   C8LL: ARX8'F
50.20   ARMD AR8VF
50.21   MDAR G8NE1
50.22   MDAR'8 EXP1
50.23   MDAR'8 HYPA1
50.24   ARMD SHF1
50.25   MDAR G8NE2
50.26   MDAR'8 EXP2
50.27   MDAR'8 HYPA2
50.30   ARMD SHF2
50.31   MDAR'8 SHF1
50.32   JPLS N8C8L
50.33   MDAR'H XP8S1
50.34   MDAE'H'N XP8S2
50.35   ANX8 MZR8
50.36   ARMD'H TEM1
50.37   ARMD'H CXP8S
50.40   ARAR'F'H
50.41   15420JH TEM1
50.42   N8OP
50.43   17440JH 1
```

```
ARMD TEM1
MDAR'H YPØS1
MDAE'H'N YPØS2
ANXØ MZRØ
ARMD'H TEM2
ARMD'H CYPØS
ARAR'F'H
15420/H TEM2
NØØP
17440/H 3
MDAE TEM1
MDAE'N'H SHPCR
JPAN .+2
JUMP CSW
MDAR ARØVF
JPAN CSW
MDAR EXP
ARMD'N EXP1                          [CØLLISIØN, SET TØ 60
ARMD'N EXP2
ARMD'Ø SHF1
ARMD'Ø SHF2
CSW: SWITCHØFF (SPH2,NØCØL)
JUMP NØCØL
CXPØS: 0                             [START ØF CALCULATE THETA
CYPØS: 0
CTHETA: JUMP .
MDAR JBLK+6
ARMD MILES
MDAR JBLK+7
ARMD THETA
TNT1: 0
TANEND: 0
MDAR MILES
ARBR'F                               [DISPLAY MILES
MDIR CTHETA
```

```
THZMAX: 37777
DELTH: 20
CTHTF: 0
THETA: 0
THETAT: 0
XTANF: 0
YTANF: 0
TANF: 0
NØCØL: MDAR TSHCR
ARMD SH1CR
ARMD SH2CR
MDAR HYPB1
JPLS .+2
JUMP .+3
MDAR TSHC2
ARMD SH1CR
MDAR HYPB2
JPLS .+2
JUMP .+3
MDAR TSHC2
ARMD SH2CR
MDAR STRF
JPAN NØSTARS
DISPLAY (FS,CO,CO,CO,STARS1)
MDAR'X NSTRS
JPLS TRYSTARS3
MDAR'L'N
1
ARMD NSTRS
DISPLAY (FS,CO,CO,CO,STARS2)
TRYSTARS3: MDAR'X STARS
JPLS NØSTARS
MDAR'L'N
2
ARMD STARS
```

```
DISPLAY (FS,CO,CO,CO,STARS3)
NØSTARS: MDAR'F TLA
ARMD PREV
MDAR'F TØRPS-1
ARMD SPTR
MDAR TLA
JPLS .+2
JUMP MSTAR
TRP1: ARMD PTR1
ARMD PTR2
ARMD PTR6
MDAS'F 2
ARMD PTR3
MDAS'F 1
ARMD PTR4
MDAR'I PTR1
ARMD TPTR
MDAR'I PTR3
ARMD TEM2
MDAE'I PTR4
ARMD'I PTR3
MDAR'I'X PTR2
JPAN TRP5
ARMD'H TEM1
MDAS TEXP
ARMD'I PTR2
ARAR'F'H
MDXØ'F 4
JPLS TRP30
MDAR'I PTR1
ARMD'I PREV
JPLS .+3
MDAR PREV
ARMD LPTR
MDAR FTA
```

```
54.1     ARMD'I PTR1
54.2     MDAR PTR1
54.3     ARMD FTA
54.4     JUMP TRP91
54.5     TRP30: MDX9'F 4
54.6     MDX0 TEM1
54.7     JPLS .+2
54.10    JUMP TRP90
54.11    JPSR RNDM2
54.12    777JH 777
54.13    TRP31: MDAE'I PTR3
54.14    MDAR'A CM11
54.15    ARMD'I'X SPTR
54.16    ARMD'I'X SPTR
54.17    MDAS'F 1
54.20    ARMD'I'X SPTR
54.21    JUMP TRP90
54.22    TRP5: MDAS'F 1
54.23    ARMD'I PTR2
54.24    MDAR SHF1
54.25    JPLS TRP51
54.26    ARX0'F
54.27    ARMD AR0VF
54.30    MDAR TEM2
54.31    MDAE'H'N XP0S1
54.32    ANX0 MZR0
54.33    ARMD TEM1
54.34    MDAR'H TEM2
54.35    MDAE'H'N YP0S1
54.36    ANX0 MZR0
54.37    MDAE TEM1
54.40    MDAE'N'H SH1CR
54.41    JPAN .+2
54.42    JUMP TRP51
54.43    MDAR AR0VF
```

```
55.1    JPAN TRP51
55.2    ARXØ'F
55.3    ARMD'I PTR2
55.4    MDAR EXP
55.5    ARMD'N EXP1
55.6    ARXØ'F
55.7    ARMD XGRV1
55.10   ARMD YGRV1
55.11   JUMP TRP90
55.12   TRP51: NØØP
55.13   MDAR SHF2
55.14   JPLS TRP52
55.15   ARXØ'F
55.16   ARMD ARØVF
55.17   MDAR TEM2
55.20   MDAE'H'N XPØS2
55.21   ANXØ MZRØ
55.22   ARMD TEM1
55.23   MDAR'H TEM2
55.24   MDAE'H'N YPØS2
55.25   ANXØ MZRØ
55.26   17440JH 1                    [ADDED TØ MAKE Y DIFF 2 TIMES AS LARGE
55.27   MDAE TEM1
55.30   MDAE'N'H SH2CR
55.31   JPAN .+2
55.32   JUMP TRP52
55.33   MDAR ARØVF
55.34   JPAN TRP52
55.35   ARXØ'F
55.36   ARMD'I PTR2
55.37   MDAR EXP
55.40   ARMD'N EXP2
55.41   ARXØ'F
55.42   ARMD XGRV2
55.43   ARMD YGRV2
```

```
56.1     JUMP TRP90
56.2     TRP52: N0ØP
56.3     JUMP TRP40
56.4     TRP41: ARMD PTR6
56.5     ARMD PTR7
56.6     MDAS'F 2
56.7     ARMD PTR8
56.10    MDAR'I'X'N PTR7
56.11    JPAN TRP40
56.12    ARX0'F
56.13    ARMD AR0VF
56.14    MDAR'I PTR8
56.15    MDAE'N TEM2
56.16    MDAR'H'I PTR8
56.17    MDAE'H'N TEM2
56.20    ANX0 HMZR0
56.21    ARAR'F'H
56.22    ANX0 HMZR0
56.23    ARAE'F'H
56.24    MDAE'N'H TTCR
56.25    JPAN TRP42
56.26    TRP40: MDAR'I PTR6
56.27    JPLS TRP41
56.30    ARX0'F
56.31    JUMP TRP31
56.32    TRP42: MDAR AR0VF
56.33    JPAN TRP40
56.34    ARX0'F
56.35    ARMD'I PTR2
56.36    MDAR'L
56.37    30000
56.40    ARMD'I PTR7
56.41    TRP90: MDAR PTR1
56.42    ARMD PREV
56.43    TRP91: MDAR TPTR
```

```
JPLS TRP1
MDAR'H ØNE
ARMD'I'X SPTR
DISPLAY (FS,CO,CO,HFS,TØRPS)
MSTAR: NØØP
MDAR STRF
JPAN DNE
MDAR'F STARS
ARMD'L
STRMVPTR: O
STARMØVE: MDAR'I'X STRMVPTR
MDAE'L
10JH -6
MDAR'A CM11
ARMD'I STRMVPTR
ARMD'X'I STRMVPTR
MDAR'Ø ØNE
ARMD'X'I STRMVPTR
MDAR'F STARSEND
MDAE'N STRMVPTR
JPAN .+2
JUMP STARMØVE
MDAR STARS2-1
MDAR'Ø EØLBIT
ARMD STARS2-1
MDAR STARS3-1
MDAR'Ø EØLBIT
ARMD STARS3-1
MDAR STARSEND-1
MDAR'Ø EØLBIT
ARMD STARSEND-1
DNE: MDAR'N STRTF
JPAN DNE1
MDBR'N ZERØ
JUMP ØUTPUT
```

```
TCBUNT: 0
DNE1: MDAR STRTF
JPAN BUTPUT
JUMP DNE1
BUTPUT: ARAR'X TCBUNT
MD07'L; 0
JPSR STBVER
NBBP
BUT1: JPSR CBFF
MD10'L'B
CLKBN
JUMP LBBP
BUT2: BUT1
BUT3: NBBP
JUMP'I BUT2
NBBP
NBBP
NBBP
MACRB1 .M (X,Y)
(X\00   77776) VH (Y\00   77776)
ENDM
MACRB1 .D (X,Y)
(X\00   77776) VH (Y\00   77776) + 1
ENDM
SHIP1:
.M (300,1)
.M (300,1)
.D (200,20)
.D (1,30)
.D (-140,25)
.D (-200,10)
.D (-200,-10)
.D (-140,-25)
.D (1,-30)
.D (200,-20)
.D (300,1)
```

```
61.1     .M (170,22)
61.2     .D (-40,140)
61.3     .D (1,30)
61.4     .M (1,-30)
61.5     .D (-40,-140)
61.6     .D (170,-22)
61.7     .M (-75,27)
61.10    .D (-140,65)
61.11    .D (-240,73)
61.12    .D (-170,11)
61.13    .M (-110,1)
61.14    .D (-240,1)
61.15    .M (-170,-11)
61.16    .D (-240,-73)
61.17    .D (-140,-65)
61.20    .D (-75,-27)
61.21    .M (-200,1)
61.22    1√H
61.23    SHIP2:
61.24    .M (10,10)
61.25    .M (10,10)
61.26    .D (-10,10)
61.27    .M (1,10)
61.30    .D (1,-10)
61.31    .M (10,-10)
61.32    .D (-10,-10)
61.33    .M (0,100)
61.34    .D (20,10)
61.35    .D (24,-30)
61.36    .D (30,-200)
61.37    .D (24,-330)
61.40    .D (14,-340)
61.41    .D (-14,-340)
61.42    .D (-24,-330)
61.43    .D (-30,-200)
```

```
62.1        .D (-24,-30)
62.2        .D (-20,10)
62.3        .D (0,100)
62.4        .M (-14,-50)
62.5        .D (-14,-34)
62.6        .D (-10,-24)
62.7        .D (10,-24)
62.10       .D (14,-34)
62.11       .D (14,-50)
62.12       .M (-20,-50)
62.13       .D (20,-50)
62.14       .D (20,-210)
62.15       .D (-20,-210)
62.16       .D (-20,-50)
62.17       .M (10,-210)
62.20       .D (10,-240)
62.21       .D (-10,-240)
62.22       .D (-10,-210)
62.23       1√H 0
62.24       SHIP1X:
62.25       .M (300,1)
62.26       .M (300,1)
62.27       .D (200,20)
62.30       .D (1,30)
62.31       .D (-140,25)
62.32       .D (-200,10)
62.33       .D (-200,-10)
62.34       .D (-140,-25)
62.35       .D (1,-30)
62.36       .D (200,-20)
62.37       .D (300,1)
62.40       .M (170,22)
62.41       .D (-40,140)
62.42       .D (1,30)
62.43       .M (1,-30)
```

```
63•1        •D (-40,-140)
63•2        •D (170,-22)
63•3        •M (-75,27)
63•4        •D (-140,65)
63•5        •D (-240,73)
63•6        •D (-170,11)
63•7        •M (-110,1)
63•10       •D (-240,1)
63•11       •M (-170,-11)
63•12       •D (-240,-73)
63•13       •D (-140,-65)
63•14       •D (-75,-27)
63•15       •M (-200,1)
63•16       1√H   0
63•17       SHTV1=•-2
63•20       SHIP2X:
63•21       •M (10,10)
63•22       •M (10,10)
63•23       •D (-10,10)
63•24       •M (1,10)
63•25       •D (1,-10)
63•26       •M (10,-10)
63•27       •D (-10,-10)
63•30       •M (0,100)
63•31       •D (20,10)
63•32       •D (24,-30)
63•33       •D (30,-200)
63•34       •D (24,-330)
63•35       •D (14,-340)
63•36       •D (-14,-340)
63•37       •D (-24,-330)
63•40       •D (-30,-200)
63•41       •D (-24,-30)
63•42       •D (-20,10)
63•43       •D (0,100)
```

```
•M (-14,-50)
•D (-14,-34)
•D (-10,-24)
•D (10,-24)
•D (14,-34)
•D (14,-50)
•M (-20,-50)
•D (20,-50)
•D (20,-210)
1√H 0
0
SHTV2=•-2
STARS: -2
STARS1: 0
STARS2: LØC •+150•
STARS3: LØC •+150•
STARSEND: LØC •+75•
LØC •+4
S0=77777;S77777=77777
S1=1444;S2=3106;S3=4544;S4=6174
S5=7615;S6=11224;S7=12620;S10=14176
S11=15535;S12=17053;S13=20347;S14=21616
S15=23040;S16=24232;S17=25373;S20=26501
S21=27554;S22=30571;S23=31550;S24=32467
S25=33345;S26=34161;S27=34733;S30=35441
S31=36102;S32=36477;S33=37025;S34=37305
S35=37517;S36=37661;S37=37754;S40=37777
AM=77777
MACRØ1 QUADRANT(VAL1,VAL2)
LØC(•+40);ENDM
MACRØ2 QUADRANT(VAL1,VAL2)
CC=0
REPEAT 40
DD=40-CC
VAL1 AM√H (VAL2 AM)
```

```
CC=CC+1
ENDR
ENDM               [REM
SNC0S:JUMP .
MDAR'A AMSK
ARMD'H DEL
17440/H 8.
MDAS'F'N 77
JPAN .+3
ARAR'N'F
MDAS'F 1
MDAE'L; MDAR TBL+77
ARIR'F
ARMD PT
MDAR DEL
JPAN SN1
MDAR'A'F'H 377
ARMD FLAG
SN0:
16440/H 12137
N00P
ARMD DEL
15420/H DEL
N00P
ARAR'N'H'F
MDAS'F 37777
ARMD F1
15420/H PT
AMSK: N00P 77777
ARMD C0S
MDAR'N DEL
15400/H PT
N00P
MDAE C0S
MDAR'A'H AMSK
```

```
ARMD'H COS
MDAR F1
15400/H PT
NOOP
ARMD SIN
MDAR DEL
15420/H PT
NOOP
MDAE SIN
MDXO'H FLAG
MDAR'A'H AMSK
ARMD'H SIN
MDIR SNCOS
SN1: MDXO'F'H 377
MDAR'A'F'H 377
ARMD'O FLAG
JUMP SNO
TBL:
QUADRANT(S\ CC,S\ DD)
QUADRANT(S\ DD,-S\ CC)
SINE:
SIN: 0
COSN:
COS: 0
F1: 0
PT: 0
DEL: 0
FLAG: 0
EXPOF: JUMP .
ARXO'F
ARMD RCOUNT1
EXPOF1: ARXO'F                  [EXPODENTIAL RANDOM NUMBER GENERATOR
ARMD RCOUNT2  [ADDITIVE TYPE
ARMD RSUM
ARAR'X RCOUNT1
```

```
67.1        JPSR RANDM
67.2        MDAR'A RAMSK
67.3        ARMD RTEST
67.4        JPSR RANDM
67.5        MDAR'A RAMSK
67.6        ARAR'X RCØUNT2
67.7        MDAS RSUM
67.10       ARMD RSUM
67.11       MDAR'H RSUM
67.12       MDAE'N RTEST
67.13       JPAN .+4
67.14       JPSR RANDM
67.15       MDAR'A RAMSK
67.16       JUMP .-10
67.17       MDAR RCØUNT2
67.20       MDAR'A'L; 1                 [TEST FØR ØDD
67.21       JPLS .+2
67.22       JUMP EXPØF1
67.23       MDAR RCØUNT1
67.24       MDAS'N'L; 1
67.25       ARAR'F'K
67.26       ARMD RCØUNT1
67.27       MDAR RTEST
67.30       17440/H 10
67.31       NØØP
67.32       MDAS RCØUNT1
67.33       MDIR EXPØF
67.34       RCØUNT1: 0
67.35       RCØUNT2: 0
67.36       RTEST: 0
67.37       RSUM: 0
67.40       RAMSK: 37777
67.41       DATASTØRE: 0
```

```
70•1        IBLK: 0                     [INIT
70•2        0
70•3        0
70•4        0
70•5        0
70•6        0
70•7        0
70•10       0
70•11       0
70•12       0
70•13       JBLK: 37777   [XPØS1
70•14       77777                       [YPØS1
70•15       40000                       [YAW1
70•16       57777                       [XPØS2
70•17       57777                       [YPØS2        DATA FRØM AGT-2
70•20       20000                       [YAW2
70•21       2000                        [RANGE
70•22       3000                        [BEARING
70•23       0                           [TØRP2F
70•24       0
70•25       TERMINATE
70•26
```

```
1.1                        C0MPUTER PR0GRAM B
1.2
1.3                  RADAR AND SHIP INF0RMATI0N SIMULAT0R
1.4
1.5                               (RADAR)
1.6
1.7        *******************************************************************
1.10
1.11        [     THIS PR0GRAM SIMULATES A REAL TIME SYSTEM,
1.12        [     A DLGN-36, USS CALIF0RNIA, AND A F-111 FIGHTER.
1.13        [     THE SIMULATI0N M0DELS A HIGH SPEED ATTACK 0N THE
1.14        [     SHIP, THE PLANE DETECTS THE SHIP AT 40 MILES
1.15        [     AND RELEASES AN ANTI SHIP MISSILE AT 20 MILES.
1.16        [     THE SHIP DETECTS THE PLANE AT 100 MILES AND
1.17        [     RELEASES 2 ADVANCED TERRIER MISSILES AS S00N
1.20        [     AS P0SSIBLE WITH THE DETERMINED DELAYS.
1.21        [     ALL SYSTEMS ARE SIMULATED WITH THE DELAYS
1.22        [     BEING CALCULATED BY THE C0MPUTER.
1.23        [
1.24        [   H.D.K.  8/12/69
1.25        [   D.A.T.  UBC  10/20/69
1.26        [   E. A. P0TTER   1/9/70   M0D 70-11-9
1.27
1.30       *******************************************************************
```

```
 EXPUNGE
TITLE RADAR
ENTRY RADAR,THRST,B0X,RANDM,CL0CK,R0TATE,INITL,STARSGEN
ENTRY INI1,L00P,1SHIP,1SH4,1SH41,1SH2,1SH6,2SHIP
ENTRY 2SH4,2SH41,2SH2,2SH6,C0LL,TRYSTARS3,TRP1,TRP5,MSTAR
ENTRY STARM0VE,SHIP1,SHIP1X,FIREC0NTL,ST0VER,DISPT,0UTPUT,DISPDATA
ENTRY EXP0F,DATAST0RE,CTHETA,THETA
ENTRY GETIB,PUTJB
MACR01 SCL; SCLXX ENDM
MACR01 L00P; L00PX ENDM
MACR01 SWITCH0FF (NN,L0C)     [T0 TEST SWITCH ACTI0NS
MDAR'N SWITH
17500/H 13.+NN                [ARLS 13. + N
N00P
JPAN L0C
ENDM
MACR01 DISPLAY (SCALE,DX,DY,DZ,L0C)
IFSAME (L0C)()                [IF NULL, D0 N0THING
ELSE
MDAR'N DISDN
JPAN .-1                      [DISPLAY D0NE
ARMD DISDN
MDAR'L
MD05 L0C                      [STARTS AVG WITH LIST (L0C)
ARMD E0VPV
ENDC
MDAR SCALE
15420/H PS                    [MULT L0WER
N00P
MDAR'L'A
77777/H 0
71100/H                       [AR11'F
26000/H DZ                    [MD06 (SET INTENSITY)
MDAR DX
15420/H PS                    [MULT L0WER
```

```
NØØP
MDAR'L'A
77777JH
ARMD DAT
MDAR DY
15420JH PS
NØØP
17440JH 15.    [ARRS 15.
NØØP
MDAR'L'A
77777
MDAR'Ø DAT
67100JH
MD10'A AVGØF   [STØPS CLØCK
MD10'Ø AVGØN   [STARTS AVG
MDIR EØVPV                      [MDDS LØC
ENDM
MD05=25000JH; MD07=27000JH; MD10=30000JH
SPL1=1; SPR1=5; SPH1=9.; SPF1=13.; SPT1=0
SPL2=16.; SPR2=12.; SPH2=8.; SPF2=4; SPT2=-1
 [CLKØN = CLØCK SETTING,  ARØPV = ØVERFLØW PIVØT
CLKØN=1000JH; FRMPV=77755; EØVPV=77756; EØLPV=77757; ARØPV=77771
THRST: 14                       [THRUST
TVC: 4                          [
HYPRA: 120.    [HYPER TIME A
TTIME: 300.    [TØRPEDØ TIME
PØFS: 1000                      [PØINT ØF FIRE START
TVEL: 110                       [TØRPEDØ VELØCITY
HYPRB: 120.    [HYPER TIME B
EXP: 60.                        [TIME FØR EXPLØSIØN
GØNE: 200.                      [TIME TØ BE GØNE
TPCT: 32.                       [TØRPEDØ CØUNT
SUNR: 600                       [SUN RADIUS
SHPCR: 25                       [SHIP CIRCLE FØR CØLISIØN
TSHCR: 770                      [TØRPEDØ TØ SHIP CIRCLE RADIUS
```

```
4.1    TSHC2: 500
4.2    TTCR: 500                              [T8RPED8 T8 T8RPED8 RADIUS
4.3    SCL: 2000                              [SCALE FACT8R F8R SHIPS
4.4    PS: 23000                              [P8SITIVE FULL BRIGHTNESS
4.5    YAW: 40                                [YAW RATE
4.6    NSTRS: -1                              [STARS 8N FLAG
4.7    STRXM: 10
4.10   SUNBM: 0
4.11   RLT: 200                               [TIME 8F RELEASE 8F T8RPS
4.12   FUEL: 2400.     [AM8UNT 8F FUEL
4.13   TEXP: 6000                             [SIZE F8R ESPL8SI8N
4.14   KGRAV: 100                             [AM8UNT 8F GRAVITY
4.15   HYRT: 120.
4.16   DAT: 0
4.17   IREPEAT Q,(1,2)
4.20   RLT\Q: 0
4.21   G8NE\Q: 0
4.22   EXP\Q: 0
4.23   HYPA\Q: 0
4.24   HYPB\Q: 0
4.25   XACC\Q: 0
4.26   YACC\Q: 0
4.27   FUEL\Q: -0
4.30   TVC\Q: -0
4.31   XVEL\Q: 0                              [TW8 LISTS 1 AND 2
4.32   YVEL\Q: 0
4.33   XGRV\Q: 0
4.34   YGRV\Q: 0
4.35   HYRT\Q: 0
4.36   TPCT\Q: 0
4.37   SHF\Q: 0
4.40   SH\Q\CR: 0
4.41   Q\PTBF: 0
4.42   RNDL\Q: REPEAT 8., 0; ENDR [8 RAND8M NUMBERS EACH
4.43   ENDI
```

```
BØX: 07000✓H 54000
07000✓H 54000
37776✓H 54001
37776✓H 40001
07000✓H 40001
07000✓H 54001
07001✓H 54001
TØRPS: 0
TØRPA: 0
YAW1: 20000
YAW2: 60000
XPØS1: 60000  [STARTING DATA FØR FIGURES
YPØS1: 60000
XPØS2: 57777
YPØS2: 57777
INIT: 0
DØNEF: 0
STRTF: 0
SUNB: -0
DISDN: -0
STPTR: STARS-1                    [STAR PØINTER
FTA: 0                            [INIT TØRPA +134
TLA: 0
TEM1: 0                           [INIT TØRPA +4
TEM2: 0
TPTR: 0
ARØVF: 0                          [AR ØVERFLØW STØRAGE
SSCL: 0
PTR1: 0
PTR2: 0                           [TØRPEDØ PØINTERS
PTR3: 0
PTR4: 0
PTR6: 0
PTR7: 0
PTR8: 0
```

```
LPTR: 0
SPTR: 0                              [INIT TØRPS -1
PREV: 0
SWITH: 0                             [SWITCH H
SWITL: 0                             [SWITCH L
NSTR: 0                              [NØ STARS
SDFLG: 0
BRCNT: 0
STTEM: 0
BØXF: 0                              [BØX FLAG
GRVF: 0                              [GRAVITY FLAG
STRF: 0
KILSN: 0
ZERØ:C0: 0
AMASK: 77777
ØNE:C01: 1
EØLBIT: 1√H 0
CM11: -(1√H 1)
SEVEN: 7
C11: 1√H 1
MZRØ: -0
HMZRØ: 77777√H
GHAØF: -20√H
FS: 37777                            [DISPLAY RANDØM NUMBER GENERATØR
HFS: 20000
AVGØF: CLKØN  [01000
AVGØN: 60740√H
RANDM: JUMP . [SIMPLE RANDØM NUMBER GENERATØR
MDAR'H RNDN
MDAE'F'B 53110
MDXØ .-1
ARMD RNDN
MDIR RANDM
RNDN: 0
RNDM2: JUMP . [THREE STEP RANDØM NØ GEN
```

```
JPSR RANDM
17500√H 5
MDAR'A'I RNDM2
ARMD RNDT
JPSR RANDM
JPSR RANDM
17500√H 7
MDAR'A'I RNDM2
MDAE RNDT
MDAE'I'N RNDM2
MDIR'X RNDM2
RNDT: 0
ARØSB: JUMP . [FØR ØVERFLØW
ARMD'Ø ARØVF
MDIR ARØSB
NULL: JUMP .  [FØR INDIRRECT JUMP
JUMP'I NULL
CLØCK: JUMP . [UPDATES CLØCK
ARMD EØLAR
MDAR'H'X DØNEF                  [TEST DØNE FLAG
JPAN .+2
ARMD'Ø STRTF  [IF YES, START FLAG
MDAR EØLAR
JUMP'I CLØCK
EØLAR: 0
EØL: JUMP .                     [END ØF LIST, SET DISPLAY DØNE
ARMD'Ø DISDN
JUMP'I EØL
CGRAV: JUMP . [CALCULATES GRAVITY
MDIR CGRAV
RØTATE: JUMP .                  [CALCULATES RØTATIØN
MDIR RØTATE
TEMRØTATE: 0
XRØTATE: 0
YRØTATE: 0
```

```
XYRØTATE: 0
RADAR: JUMP . [ENTRY PØINT
MDAR'I RADAR
JPLS .+2
JUMP INITL-4
ARMD THRST
MDAR'I'X RADAR
ARMD TVEL
MDAR'X'I RADAR
ARMD KGRAV
MDAR'L; 57777 [SET INITIAL PØSITIØN ØF SHIP
ARMD XPØS2
ARMD YPØS2
INITL: MDAR'F NULL              [START ØF INITIALIZATIØN
ARMD FRMPV                      [START CLØCK
MD10 ZERØ
MDAR'F EØL
ARMD EØLPV
MDAR'L
JPSR ARØSB
ARMD ARØPV
MDAR'F STARS
ARMD'L
STARSPTR: 0
STARSGEN: NØØP
ARXØ'F                          [SET CØRE TØ ZERØ
ARMD RLT1
ARMD GØNE1
ARMD EXP1
ARMD HYPA1
ARMD HYPB1
ARMD XVEL1
ARMD YVEL1
ARMD XGRV1
ARMD YGRV1
```

```
11.1        ARMD HYRT1
11.2        ARMD RNDL1
11.3        ARMD RNDL1+1
11.4        ARMD RNDL1+2
11.5        ARMD RNDL1+3
11.6        ARMD RNDL1+4
11.7        ARMD RNDL1+5
11.10       ARMD RNDL1+6
11.11       ARMD RNDL1+7
11.12       ARMD'Ø TVC1
11.13       ARMD RLT2
11.14       ARMD GØNE2
11.15       ARMD EXP2
11.16       ARMD HYPA2
11.17       ARMD HYPB2
11.20       ARMD XVEL2
11.21       ARMD YVEL2
11.22       ARMD XGRV2
11.23       ARMD YGRV2
11.24       ARMD HYRT2
11.25       ARMD RNDL2
11.26       ARMD RNDL2+1
11.27       ARMD RNDL2+2
11.30       ARMD RNDL2+3
11.31       ARMD RNDL2+4
11.32       ARMD RNDL2+5
11.33       ARMD RNDL2+6
11.34       ARMD RNDL2+7
11.35       ARMD TCØUNT
11.36       ARMD FCS2F
11.37       ARMD IBLK+10  [TØRP2F
11.40       ARMD SIMF2
11.41       ARMD SIMF3
11.42       ARMD SIMF4
11.43       ARMD SIMF5
```

```
12.1        ARMD SIMF6
12.2        ARMD SIMF7
12.3        ARMD SIMF8
12.4        ARMD SIMF9
12.5        ARMD SIMF10
12.6        ARMD SIMF11
12.7        ARMD SIMF12
12.10       ARMD SIMF13
12.11       ARMD SIMF14
12.12       ARMD SIMF15
12.13       ARMD SIMF16
12.14       ARMD SIMF17
12.15       ARMD SIMF18
12.16       ARMD DATATEMP
12.17       ARMD'8 TVC2
12.20       ARMD JBLK                  [VICE INIT
12.21       ARMD TLA
12.22       ARMD T8RPA
12.23       ARMD STRTF
12.24       ARMD'8 DISDN
12.25       ARMD'8 SUNB
12.26       MDAR'L
12.27       20000
12.30       ARMD YAW2
12.31       MDAR'L; 40000
12.32       ARMD YAW1
12.33       MDAR'L; 37777
12.34       ARMD XP8S1
12.35       MDAR'L; 77777
12.36       ARMD YP8S1
12.37       MDAR'L; -20
12.40       ARMD XVEL1
12.41       MDAR'L; 1
12.42       ARMD YVEL2
12.43       ARMD SIMF1                 [ADDED
```

```
13.1      MDAR'L; 3000
13.2      ARMD THETA
13.3      MDAR SIMTEN
13.4      ARMD'N SIMTIME
13.5      MDAR'N TPCT
13.6      ARMD TPCT1
13.7      ARMD TPCT2
13.10     MDAR'N FUEL
13.11     ARMD FUEL1
13.12     ARMD FUEL2
13.13     MDAR'L
13.14     TØRPA+4
13.15     ARMD TEM1
13.16     JPSR STØVER                     [FØR SETUP
13.17     INI1: NØØP
13.20     MDAR'F CLØCK
13.21     ARMD FRMPV
13.22
13.23     MDAR'H'X NAMEC
13.24     JPAN STN4
13.25     MDAR'L; -400
13.26     ARMD DØNEF
13.27     MDAR .-2
13.30     ARMD NAMEC
13.31     STN2: JPSR CØFF
13.32     JPSR SUP
13.33     STNAM
13.34     STNAM
13.35     NAME
13.36     MDAR'H'X NAMEC
13.37     JPAN STN2
13.40     JPSR CØN
13.41     JUMP .
13.42     STNAM: NØØP
13.43     JUMP'I STN1
```

```
STN1: .+1
MDAR'H'X D8NEF
JPAN STN2-2
JPSR C8FF
MDAR'L
MTEN: -10.
ARMD NAMEC

STN4: MD10'8'L
CLK8N

JUMP STN3
NAMEC:-0
NAME:
N8CARRET
ASCII '∆N∆N∆N∆N∆E∆E∆C∆F∆X∆060∆Y∆070∆' RADAR ∆'
∆C∆F∆F∆X∆050∆S∆N A C8MBAT WEAP8N ∆C∆F∆N
∆X∆050∆N∆N SYSTEM SIMULATI8N  ∆C∆F∆F∆N
∆X∆064∆N∆N∆S WRITTEN BY ∆C∆E∆F
∆X∆074∆N∆N LT E. A. P8TTER ∆C∆F∆S
∆X∆100∆N∆N JUNE 1970, NPGS ';
CARRET
STN3: ARMD'8 D8NEF             [SET UP C8UNTER
JPSR RANDM                     [GET RAND8M NUMBERS
MDAR'N STRTF  [WAIT UNTIL FRAME INTERRUPT
JPAN .-2
L88P: MDAR'F 77775             [8VERALL L88P 8F PR8GRAM
ARMD D8NEF                     [SETS UP C8UNTER
MDAR'H JBLK                    [TEST F8R END 8F GAME
JPAN INITL
ARX8'F
ARMD STRTF
MDAR SWITH
ARMD SWITL
MDIC'8'L
```

```
40√H
S5AR'F'H                              [READS BUTTON ACTIONS INTO AR
17500√H 1
MDAR'L'A
1√H 77776
ARMD SWITH                            [PUTS IT INTO SWITH
MDIC'A'L
-40√H
S5MD TEM1
MDAR'H TEM1
MDAR'A ONE
MDAR'O SWITH
ARMD SWITH
MDAR TEM1
MDAR'A'H'L
6
MDAR'O SWITH
ARMD SWITH
MDAR SWITL
MDXO'A SWITH
ARMD TEM2
MDAR'H SWITH
JPAN INITL                            [TESTS FOR NEW GAME
17500√H 1
NOOP
JUMP DOBOX
DISPT: JUMP .
ARMD NUMBER
MDAR'N'L; 4
ARMD BCOUNTER [DISPLAYS TCOUNT
MDAR ONETWOF
JPLS .+3
MDAR'F NUMBER1-1
JUMP .+2
MDAR'F NUMBER2-1
```

```
ARMD STØLAC                     [INITILIZE TØ NUMBER -1
MDAR NUMBER
ARMD SAVEN
TESTC: MDAR SAVEN
17500JH 3                       [SHIFT 3 LEFT FØR 1 ST NUMBER
ARMD SAVEN
ARAR'H'F
JPLS INSERTN  [ALSØ TESTS FØR LEADING BLANKS
MDAR BLANK
INSERT: ARMD'I'X STØLAC
MDAR'N'X'H BCØUNTER             [TESTS FØR END ØF NUMBERS
JPAN DISEND                     [DØNE
JUMP TESTC                      [SHIFT 1 LEFT FØR NUMBER
INSERTN: MDAR'A NMASK           [GET ØNLY NUMBER
MDAR'Ø SIXTY  [ADD IN 60
ARAR'B'F                        [SHIFT 1 LEFT FØR CØRRECT PØSIT
JUMP INSERT
DISEND: MDAR STØLAC
MDAS'F'N 1
ARMD STØLAC
MDAR'I STØLAC
MDAR'Ø DECPT
ARMD'I STØLAC [ADD IN A DEC PT TØ MAKE MILES
MDIR DISPT
STØLAC: 0
BCØUNTER: 0
BLANK: 100
SIXTY: 60
NMASK: 7
DECPT: 134JH
SAVEN: 0
NUMBER: 0
ØNETWØF: 0
DISPDATA: JUMP .
JPSR FIRECØNTL
```

```
17.1      JUMP .
17.2      DISP0: 0
17.3      ARXØ'F
17.4      ARMD ØNETWØF
17.5      MDAR MILES
17.6      17500JH 2
17.7      JPSR DISPT
17.10     MDAR'F 1
17.11     ARMD ØNETWØF
17.12     MDAR THETA                  [CHANGED FRØM TCØUNT
17.13     17500JH 2
17.14     JPSR DISPT
17.15     DPEND: JPSR FC2
17.16     JUMP .
17.17     DISP9: NØØP
17.20     MDIR DISPDATA [RETURN
17.21     DISP8: NØØP
17.22     JUMP'I DISP7
17.23     DISP7: DISP9
17.24     DISP1: DISP0
17.25     DISP2: NØØP
17.26     JUMP'I DISP1
17.27     MILES: 65432  [SET UP INIT FØR TEST
17.30     TEMPDISP: 0
17.31     DATATEMP: 0
17.32     STØVER: JUMP .
17.33     ARAR'X SIMTIME
17.34     ARAR'X SIMTIME2
17.35     MDAR SIMF1
17.36     JPLS .+2
17.37     JUMP DISWD1
17.40     MDAR'H SIMTIME
17.41     JPAN DISWD1                 [DELAY ØF 10 TIME UNITS
17.42     MDAR'N SIMTEN
17.43     ARMD SIMTIME
```

```
ARMD SIMTIME2
JPSR CTHETA
MDAR'N'H MILES
MDAE'L; 2700JH                  [100 MILES
JPAN  DISWD1
ARX0'F
ARMD SIMF1
MDAR FLG0NE
ARMD SIMF2
ARMD SIMF3
ARMD SIMF4
ARMD THET2F
MDAR YAW1
MDAS'N THETA
ARMD'H THET2
DISWD1: N00P
MDAR'F IBLK                     [TRANSFERS DATA T0 IBLK
ARMD IBLKF
MDAR XP0S1
ARMD'I IBLKF
MDAR YP0S1
ARMD'I'X IBLKF
MDAR YAW1
ARMD'I'X IBLKF
MDAR XP0S2
ARMD'I'X IBLKF
MDAR YP0S2
ARMD'I'X IBLKF
MDAR YAW2
ARMD'I'X IBLKF
MDAR MILES
ARMD'I'X IBLKF
MDAR THETA
ARMD'I'X IBLKF

DISWD2: N00P  [WRITES AND READS DATA FR0M AGT-1
```

```
MDAR'F WD4
JPSR DISPDATA [TEST
MDAR'H SIMTIME
JPAN .+4
MDAR'N DISPTIME                 [100
ARMD SIMTIME
JPSR CTHETA

JPSR $RθWFW                     [WRITE θUT IBLK
          -0
         115
      IBLK
GETIB: 10.

JPSR $RθWFW                     [READ IN JBLK
           0
         116
       JBLK
PUTJB: 10.
JPSR $FINSH
DISWD3: MDAR SIMF4
JPLS .+2
JUMP DISWD4
MDAR YTANF
JPLS .+5
MDAR'H YAW1
MDAE THET2
JPAN DISWD4
JUMP .+4
MDAR'H'N YAW1
MDAE'N THET2
JPAN DISWD4
ARXθ'F
ARMD THET2F
ARMD SIMF4
```

```
MDAR FLGØNE
ARMD SIMF5
MDAR YAW1
ARMD'H THET2
DISWD4: MDAR SIMF5
JPLS .+2
JUMP DISWD5
ARXØ'F                                  [IF NØT WITHIN RANGE
ARMD IBLK+10  [DØNT FIRE
MDAR'N'H MILES
MDAE'L; 1000/H
JPAN .+10
MDAR'H YAW1                             [TEST TØ SEE IS STILL
MDAE'N THET2  [ØN CØURSE
ANXØ MZRØ
ARAR'H'F
JPLS .+3
MDAR FLGØNE
ARMD IBLK+10  [FIRE
JUMP ENDØVER
DISWD5: NØØP
DISWD6: NØØP
DISWD7: NØØP
DISWD8: NØØP
DISWD9: NØØP
DISWD10: NØØP
DISWD11: NØØP
DISWD12: NØØP
DISWD13: NØØP
DISWD14: NØØP
DISWD15: NØØP
DISWD16: NØØP
DISWD17: NØØP
ENDØVER: MDIR STØVER
SIMF1: 0
SIMF2: 0
```

```
23.1         SIMF3: 0
23.2         SIMF4: 0
23.3         SIMF5: 0
23.4         SIMF6: 0
23.5         SIMF7: 0
23.6         SIMF8: 0
23.7         SIMF9: 0
23.10        SIMF10: 0
23.11        SIMF11: 0
23.12        SIMF12: 0
23.13        SIMF13: 0
23.14        SIMF14: 0
23.15        SIMF15: 0
23.16        SIMF16: 0
23.17        SIMF17: 0
23.20        SIMF18: 0
23.21        FCS2F: 0
23.22        THET2F: 0
23.23        THET2: 0
23.24        FLGØNE: 1
23.25        SIMTEN: 10.
23.26        SIMTIME: 0
23.27        SIMTIME2: 0
23.30        DISPTIME: 100.
23.31        FIRECØNTL: JUMP .
23.32        ARMD CHRSTRING
23.33        JPSR CØFF
23.34        JPSR SUP
23.35        DISP2
23.36        DISP2
23.37        CHRSTRING: NØØP
23.40        JPSR CØN
23.41        MDIR FIRECØNTL
23.42        FC2: JUMP .
23.43        ARMD FIRESAVE
```

```
24.1      MD07'L; 0
24.2      JPSR CØFF
24.3      JPSR SUP
24.4      DISP8
24.5      DISP8
24.6      NUMBER1-3
24.7      JPSR CØN
24.10     MDAR FIRESAVE
24.11     MDIR FC2
24.12     CØFF: JUMP .
24.13     MDIC'A'L
24.14     -10
24.15     MD10'A'L
24.16     776√H
24.17     MDIR CØFF
24.20     CØN: JUMP .
24.21     MD10'B'L
24.22     776√H
24.23     MDIC'B'L
24.24     10
24.25     MDIR CØN
24.26     SUP: JUMP .
24.27     MDAR'I SUP
24.30     ARMD 77736
24.31     MDAR'X SUP
24.32     MDAR'I SUP
24.33     ARMD 77737
24.34     MDAR'X SUP
24.35     MDAR'I SUP
24.36     ARMD 77735
24.37     MDAR'X SUP
24.40     MDIR SUP
24.41     FIRESAVE: 0
24.42     MD06= 26000√H
24.43     AR07= 27100√H
24.44     MD11= 31000√H
```

```
25.1        WD4: 0              [TRACKING
25.2        0463005670
25.3        1743000000
25.4        5224440606
25.5        4562347216
25.6
25.7        0
25.10       0464005700
25.11       1742000000
25.12       NUMBER1: 0
25.13       0
25.14       0
25.15       0
25.16       0
25.17       2023244630                    [ MILES, TIME = .
25.20       4264600000
25.21       0463005710                    [NEW X,Y VALUES
25.22       5222246612
25.23       2017220000
25.24       NUMBER2: 0
25.25       0
25.26       0
25.27       0
25.30       0
25.31       2024642606                    [ SEC
25.32       0000100000   [END OF DATA
25.33       DOBOX: NOOP
25.34       1SHIP: NOOP
25.35       ARXO'F
25.36       ARMD XACC1
25.37       ARMD YACC1
25.40       MDAR SCL
25.41       ARMD SSCL
25.42       MDAR RLT1
25.43       JPLS .+2
25.44       JUMP .+2
```

```
ARAR'X RLT1
MDAR GØNE1
JPLS 1SH5
MDAR EXP1
JPLS 1SH4
MDAR HYPA1
JPLS 1SH6
MDAR HYPB1
JPLS 1SH7
SWITCHØFF (SPR1,TURNR1)                          [THIS IS THE TURN RØUTINE
MDAR YAW1
MDAS'N YAW
ARMD YAW1
TURNR4: JPSR SNCØS
MDAR'L;PTEN: 20
15420JH CØSN
NØØP
ARAR'H
ARMD XVEL1
MDAR PTEN
15420JH SINE
NØØP
ARAR'H
ARMD YVEL1
JUMP TURNR3
TURNR1: MDAR THET2F
JPLS .+2
JUMP .+4
MDAR YTANF
JPLS TURNR4-3
JUMP TURNR5
SWITCHØFF (SPL1,TURNR2)
TURNR5: MDAR YAW1
MDAS YAW
ARMD YAW1
```

```
27.1    JUMP TURNR4
27.2    TURNR2: MDAR YAW1
27.3    JPSR SNCØS
27.4    TURNR3: SWITCHØFF (SPT1,1NEXT) [END ØF TURN
27.5    1NEXT: NØØP
27.6    1SH4: MDAR XPØS1
27.7    MDAS XVEL1
27.10   ARMD XPØS1
27.11   MDAR YPØS1
27.12   MDAS YVEL1
27.13   ARMD YPØS1
27.14   MDAR HYRT1
27.15   JPLS .+2
27.16   JUMP .+2
27.17   ARAR'X HYRT1
27.20   MDAR EXP1
27.21   JPLS 1SH3
27.22   1SH10: NØØP
27.23   1SH40: NØØP
27.24   1SH42: NØØP
27.25   1SH41: NØØP
27.26   1SH11: NØØP
27.27   1SH2: NØØP
27.30   SWITCHØFF (SPF1,1SH20)
27.31   1SH2P: MDAR TPCT1
27.32   MDAR TPCT1
27.33   JPLS .+2
27.34   JUMP 1SH20
27.35   MDAR RLT1
27.36   JPLS 1SH20
27.37   MDAR FLGØNE
27.40   ARMD IBLK+10  [FØR MANUAL FIRE, TØRP2F
27.41   MDAR RLT
27.42   ARMD'N RLT1
27.43   JUMP 1SH99
```

```
1SH20: N0ØP
SHIP1XPTR: 0
SHIP1PTR: 0
SHIP1RØT: NØØP
1SH20A: NØØP
1SH21: NØØP
1SH5: NØØP
1SH6: NØØP
1SH7: NØØP
1SH30: NØØP
1SH14: NØØP
1SH3: NØØP
JUMP 1SH99
SSCL2: 10000
YAWC2: 100
1SH99:
2SHIP: NØØP
ARXØ'F
ARMD XACC2
ARMD YACC2
MDAR RLT2
JPLS .+2
JUMP .+2
ARAR'X RLT2
MDAR GØNE2
JPLS 2SH5
MDAR EXP2
JPLS 2SH4
MDAR HYPA2
JPLS 2SH6
MDAR HYPB2
JPLS 2SH7
SWITCHØFF (SPR2,.+4)
MDAR YAW2
MDAS'N YAWC2
ARMD YAW2
```

```
SWITCHOFF (SPL2,.+4)
MDAR YAW2
MDAS YAWC2
ARMD YAW2
MDAR YAW2
JPSR SNCOS
MDAR'X TVC2
JPLS 2SH4
MDAR TVC
ARMD'N TVC2
2SH4: MDAR'H XVEL2
MDAE XACC2
ARMD'H XVEL2
MDAR'H YVEL2
MDAE YACC2
ARMD'H YVEL2
MDAR XPOS2
MDAS XVEL2
ARMD XPOS2
MDAR YPOS2
MDAS YVEL2
ARMD YPOS2
MDAR HYRT2
JPLS .+2
JUMP .+2
ARAR'X HYRT2
MDAR EXP2
JPLS 2SH3
JUMP 2SH99
2SH10: NOOP
2SH40: NOOP
2SH42: NOOP
2SH41: NOOP
2SH11: NOOP
2SH2: NOOP
```

```
2SH2P: NØØP
2SH20: NØØP
SHIP2XPTR: 0
SHIP2PTR: 0
SHIP2RØT: NØØP
2SH20A: NØØP
2SH21: NØØP
2SH5: NØØP
2SH6: NØØP
2SH7: NØØP
2SH30: NØØP
2SH14: NØØP
2SH3: NØØP
2SH99: 0
CØLL: NØØP
CSW: SWITCHØFF (SPH2,NØCØL)
JUMP NØCØL

CXPØS: 0                        [PRØGRAM CALCULATE THETA
CYPØS: 0
CTHETA: JUMP .
ARXØ'F              [ZERØ ØUT FLAGS
ARMD XTANF
ARMD YTANF
ARMD TANF

MDAR'H XPØS1
17440JH 3
ARMD'H CXPØS
MDAR'H XPØS2
17440JH 3
MDAE'N'H CXPØS
ARMD'H'N            CXPØS
MDAR'H YPØS1
17440JH 3
```

```
33.1      ARMD'H CYP0S
33.2      MDAR'H YP0S2
33.3      17440JH 3
33.4      MDAE'N'H CYP0S
33.5      ARMD'H'N CYP0S
33.6      MDAR'H'N CXP0S                  [TEST F0R X NEG
33.7      JPAN .+3
33.10     MDAR FLG0NE
33.11     ARMD XTANF
33.12     MDAR'H'N CYP0S                  [TEST F0R Y NEG
33.13     JPAN .+3
33.14     MDAR FLG0NE
33.15     ARMD YTANF
33.16     MDAR'H CYP0S                    [MAKE P0S
33.17     ANX0 MZR0
33.20     ARMD'H CYP0S
33.21     MDAR'H CXP0S
33.22     ANX0 MZR0
33.23     ARMD'H CXP0S
33.24
33.25     MDAR'H CYP0S
33.26     MDAE'H'N CXP0S
33.27     JPAN .+8.
33.30     MDAR FLG0NE
33.31     ARMD TANF
33.32     MDAR'H CXP0S
33.33     16420JH CYP0S                   [X2-X1/Y2-Y1
33.34     N00P
33.35     ARMD TANT
33.36     JUMP .+6
33.37     MDAR'H CYP0S
33.40     16420JH CXP0S                   [Y2-Y1/X2-X1
33.41     N00P
33.42     ARMD'L
33.43     TANT: 0
```

```
34•1        ARAR'H'N'F                          [TEST FØR DIV ERRØRS
34•2        JPAN •+2
34•3        ARMD'H TANT
34•4
34•5        TNT1: MDAR THETAT
34•6        JPSR SNCØS
34•7        MDAR    CØSN                         [TEST FØR CØSN = 0
34•10       JPLS    •+2
34•11       JUMP    TANEND
34•12       MDAR SINE                            [TEST FØR SIN = 0
34•13       JPLS •+2
34•14       JUMP TANEND
34•15       MDAR'H SINE
34•16       16420JH CØSN                         [SINE/CØSN
34•17       NØØP
34•20       MDAS'N TANT
34•21       MDAS'N'L
34•22       DELTAN: 200
34•23       JPLS •+2
34•24       JUMP TANEND                          [TEST FØR DELTAN = 0
34•25       ARAR'H'F
34•26       JPAN •+5
34•27       MDAR THETAT                             [THETA TØ LARGE
34•30       MDAS'N DELTH
34•31       ARMD THETAT
34•32       JUMP TNT1
34•33       ARAR'H'F
34•34       MDAS DELTAN                          [THETA TØ SMALL
34•35       MDAS DELTAN
34•36       ARAR'H'N'F
34•37       JPAN •+5
34•40       MDAR THETAT
34•41       MDAS DELTH                           [SINE THETA IN SINE
34•42       ARMD THETAT
34•43       JUMP TNT1
```

```
TANEND: MDAR SINE               [TEST FØR SIN = 0
JPLS .+4
ARXØ'F
ARMD THETAT
JUMP .+5
MDAR CØSN
JPLS .+3
MDAR THZMAX
ARMD THETAT

MDAR THETAT
ARMD THETA

MDAR TANF                       [TEST FØR TAN GT 45
JPLS .+2
JUMP .+4
MDAR HFS                        [20000
MDAS'N THETA
ARMD THETA
MDAR XTANF                      [TEST FØR NEG X
JPLS .+2
JUMP .+4
MDAR FS
MDAS'N THETA
ARMD THETA
MDAR YTANF                      [TEST FØR Y NEG
JPLS .+2
JUMP .+3
MDAR'H'N THETA
ARMD'H THETA


MDAR THETA
JPSR SNCØS
MDAR CØSN
JPLS .+2
```

```
36•1        JUMP •+5
36•2        MDAR'H SINE
36•3        JPAN •+10
36•4        ARAR'H'F
36•5        JPLS •+4
36•6        MDAR CXPØS
36•7        MDAS CYPØS
36•10       JUMP •+6
36•11       MDAR'H CYPØS                 [FØUND A GØØD THETA
36•12       JUMP •+2
36•13       MDAR'H'N CYPØS
36•14       16420JH SINE                 [Y/SIN THETA = Z
36•15       NØØP
36•16       17440JH 1                    [SHIFT 1 RIGHT
36•17       MDAR'A THZMAX
36•20       ARMD MILES
36•21
36•22       MDAR MILES
36•23       ARBR'F                         [DISPLAY MILES
36•24       MDIR CTHETA
36•25       THZMAX: 37777
36•26       DELTH: 20
36•27       CTHTF: 0
36•30       THETA: 0
36•31       THETAT: 0
36•32       XTANF: 0
36•33       YTANF: 0
36•34       TANF: 0
36•35       NØCØL: NØØP
36•36       TRYSTARS3: NØØP
36•37       NØSTARS: NØØP
36•40       TRP1: NØØP
36•41       JUMP TRP91
36•42       TRP30: NØØP
36•43       TRP31: NØØP
```

```
TRP5: N0OP
TRP51: N0OP
TRP52: N0OP
TRP41: N0OP
TRP40: N0OP
TRP42: N0OP
TRP90: N0OP
TRP91: N0OP
MSTAR: N0OP
STRMVPTR: 0
STARM0VE: N0OP
DNE: MDAR'N STRTF
JPAN DNE1
MDBR'N ZER0
JUMP 0UTPUT
TC0UNT: 0
DNE1: MDAR STRTF
JPAN 0UTPUT
JUMP DNE1
0UTPUT: ARAR'X TC0UNT
MD07'L; 0
JPSR ST0VER
0UT1: N0OP
JPSR C0FF
MD10'L'0
CLK0N
JUMP L0OP
0UT2: 0UT1
0UT3: N0OP
JUMP'I 0UT2
SHIP1:
SHIP2:
SHIP1X:
SHIP2X:
STARS: -2
```

```
STARS1: 0
STARS2: 0
STARS3: 0
STARSEND: 0

S0=77777;S77777=77777
S1=1444;S2=3106;S3=4544;S4=6174
S5=7615;S6=11224;S7=12620;S10=14176
S11=15535;S12=17053;S13=20347;S14=21616
S15=23040;S16=24232;S17=25373;S20=26501
S21=27554;S22=30571;S23=31550;S24=32467
S25=33345;S26=34161;S27=34733;S30=35441
S31=36102;S32=36477;S33=37025;S34=37305
S35=37517;S36=37661;S37=37754;S40=37777
AM=77777
MACR01 QUADRANT(VAL1,VAL2)
LOC(.+40);ENDM
MACR02 QUADRANT(VAL1,VAL2)
CC=0
REPEAT 40
DD=40-CC
VAL1 AM/H (VAL2 AM)
CC=CC+1
ENDR
ENDM    [REM

SNCOS:JUMP .
MDAR'A AMSK
ARMD'H DEL
17440/H 8.
MDAS'F'N 77
JPAN .+3
ARAR'N'F
MDAS'F 1
MDAE'L; MDAR TBL+77
ARIR'F
```

```
41.1        ARMD PT
41.2        MDAR DEL
41.3        JPAN SN1
41.4        MDAR'A'F'H 377
41.5        ARMD FLAG
41.6        SNO:
41.7        16440/H 12137
41.10       NØØP
41.11       ARMD DEL
41.12       15420/H DEL
41.13       NØØP
41.14       ARAR'N'H'F
41.15       MDAS'F 37777
41.16       ARMD F1
41.17       15420/H PT
41.20       AMSK: NØØP 77777
41.21       ARMD CØS
41.22       MDAR'N DEL
41.23       15400/H PT
41.24       NØØP
41.25       MDAF CØS
41.26       MDAR'A'H AMSK
41.27       ARMD'H CØS
41.30       MDAR F1
41.31       15400/H PT
41.32       NØØP
41.33       ARMD SIN
41.34       MDAR DEL
41.35       15420/H PT
41.36       NØØP
41.37       MDAE SIN
41.40       MDXØ'H FLAG
41.41       MDAR'A'H AMSK
41.42       ARMD'H SIN
41.43       MDIR SNCØS
```

```
SN1: MDX0'F'H 377
MDAR'A'F'H 377
ARMD'0 FLAG
JUMP SNO
TBL:
QUADRANT(S\ CC,S\ DD)
QUADRANT(S\ DD,-S\ CC)
SINE:
SIN: 0
C0SN:
C0S: 0
F1: 0
PT: 0
DEL: 0
FLAG: 0
EXP0F: N00P
EXP0F1: N00P
RC0UNT1: 0
RC0UNT2: 0
RTEST: 0
RSUM: 0
RAMSK: 37777
```

```
43.1    IBLKF: 0
43.2    IBLK: 0                         [XP0S1
43.3    0                                           [YP0S1
43.4    0                                           [THETA1
43.5    0                                           [XP0S2
43.6    0                                           [YP0S2
43.7    0                                           [THETA2
43.10   0                                           [MILES
43.11   0                                           [THETA
43.12   0                                           [T0RP2F
43.13   0
43.14   JBLK: 0                                     [INIT
43.15   0
43.16   0
43.17   0
43.20   0
43.21   0
43.22   0
43.23   0
43.24   0
43.25   0
43.26   DATAST0RE: 0
43.27   TERMINATE
```

```
C                       COMPUTER  PROGRAM  C
C
C                       THE MISSILE MOUNT
C
C*************************************************************************
C
C        THIS PROGRAM SIMULATES A DIGITAL FIRE CONTROL SYSTEM USING AN
C        ANALOG COMPUTER SIMULATION OF A LARGE MISSILE MOUNT.
C
C        THE DIGITAL FIRE CONTROL SYSTEM USES SIX (6) PHASES TO CONTROL
C        THE ACTION OF THE MISSILE MOUNT.  THEY ARE AS FOLLOWS:
C
C                  PHASE 0:   ACCELERATE MOUNT, START SPEEDUP
C                  PHASE 1:   DEACCELERATE MOUNT, CONTINUE SPEEDUP
C                  PHASE 2:   ZERO ACCELERATION,  MAXIMUM SPEED REACHED
C                  PHASE 3:   DEACCELERATE MOUNT, START SLOW DOWN
C                  PHASE 4:   ACCELERATE MOUNT, CONTINUE SLOW DOWN
C                  PHASE 5:   SWITCH INTO LINEAR MODE
C
C        THE SENSE SWITCHES ARE USED TO ENTER APPROPRIATE VALUES INTO
C        THE COMPUTER SYSTEM TO OPTIMIZE THE DIGITAL SYSTEM TO THE
C        ANALOG MISSILE MOUNT.
C
C        SENSE SWITCH 5 IS USED TO PRINT CONTINUOUSLY PERTINENT VALUES
C        IN THE SIMULATION.
C        SENSE SWITCH 6 IS USED TO ONLY PRINT THESE SAME VALUES DURING
C        PHASE CHANGES.
C
C*************************************************************************
C
C
C
C        THE FOLLOWING PROGRAM SEGMENT IS USED TO SETUP THE INTERFACE
C        BETWEEN THE ADAGE SYSTEM AND THE XDS - 9300 COMPUTER
C
```

```
C     LET IBLK BE DATA FØR AGT1
C     LET JBLK BE DATA FØR AGT2
      DIMENSIØN IBLK( 64),JBLK( 64)
      IER = 0
      ICØM = 15
      DØ 1000 I = 1, 64
      JBLK(I) = 5
 1000 IBLK(I) = 0
      PRINT 1001
 1001 FØRMAT (1H1)
 1010 PRINT 1101, (IBLK(I),I=1, 10),IDEV
      PRINT 1101, (JBLK(I),I=1, 10),IDEV
 1101 FØRMAT(10(1X,Ø8),I10)
 1102 FØRMAT(10(1X,Ø8),I10/)
      PAUSE
      IDEV = 1
 1003 CALL AGTCØM(IDEV,ICØM,IER,IBLK,10 ,JBLK,10 )
      PRINT 1200,IER
 1200 FØRMAT (// '   THE ERRØR IS ØF TYPE',I2,//)
      PRINT 1101, (JBLK(I),I=1, 10),IDEV
      PRINT 1101, (IBLK(I),I=1, 10)
      IF (IER .NE. 0) GØ TØ 1099
 1002 IDEV = 2
      CALL AGTCØM(IDEV,ICØM,IER,IBLK,10 ,JBLK,10 )
      PRINT 1200,IER
      PRINT 1101, (IBLK(I),I=1, 10)
      PRINT 1101, (JBLK(I),I=1, 10),IDEV
      IF (IER .NE. 0) GØ TØ 1099
 1004 CØNTINUE
S      LDA     =100000
      CALL DELAY
      PRINT 1101, (IBLK(I),I=1, 10)
      PRINT 1102, (JBLK(I),I=1, 10)
 1099 PRINT 1199
 1199 FØRMAT(//,3X, ' AN ERRØR HAS BEEN MADE' //)
```

```
C
C
      DIMENSION A(25)
      REAL K1,K2
      LOGICAL IFLAG
      NAMELIST   IREP,THR,K1,K2,SW1,SW2,SW3,SW4,SW5
C        READ IN DATA CARDS FOR POTSET
      READ 100,(A(I),I=1,20)
  100 FORMAT(10F8.4)
C        SET APPROPREATE POTS
      CALL SETPOT (4HP001,A(1),4HP002,A(2), 4HP003,A(3) ,4HP004,A(4))
      CALL SETPOT (4HP005,A(5),4HP006,A(6), 4HP007,A(7) ,4HP010,A(8))
      CALL SETPOT (4HP011,A(9),4HP012,A(10),4HP013,A(11),4HP014,A(12))
      CALL SETPOT (4HP015,A(13),4HP016,A(14),4HP017,A(15),4HP024,A(20))
      CALL SETPOT (4HP020,A(16),4HP021,A(17),4HP022,A(18),4HP023,A(19))
      E = .001
C              K1 IS THE VALUE FOR K
      K1 = 2.5
      AA= K1/25.
C        SET VARIABLE POT, DEPENDS UPON K
      CALL SETPOT(4HP047,AA)
      K2 = .9999
      SW1 = 0.
      SW2 = .9
      SW5 = -.0
      IREP =2000
      THR = .5
    1 IF (SENSE SWITCH 1) 10,2
    2 IF (SENSE SWITCH 2) 30,3
    3 IF (SENSE SWITCH 3) 30,4
    4 IF (SENSE SWITCH 4) 40,5
    5 CALL HOLD
      S2 = SW2
      SW3 = THR - .4*THR
      SW4 = THR - .2*THR
```

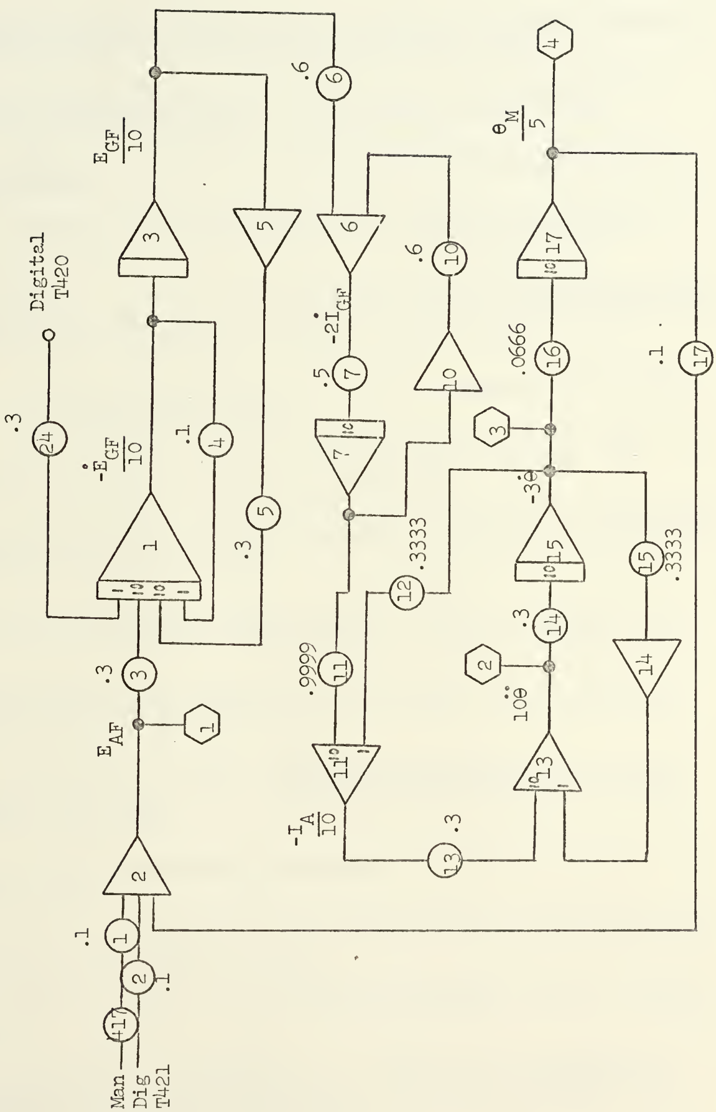
```
      THP = -K2
      IFLP = 1
      IPHASE = 0
       CHG = THR
      ICØUNT=0
      IF (SENSE SWITCH 5) 82,83
   82 PRINT 200
  200 FØRMAT (1H1)
   83 IF (SENSE SWITCH 6) 84,85
   84 PRINT 201
  201 FØRMAT (///)
   85 CALL RESET(1000)
C        SETS UP INTERRUPT SYSTEM
      CØNNECT (052,ENDSTP(IFLAG))
      IFLAG = .TRUE.
      CALL CØMPUTE
C        TEST(13) IS AN ANALØG INTERRUPT
  121 IF (TEST(13) .GT. 0) GØ TØ 121
      CALL ENABLE
  112 IF (IFLAG) GØ TØ 112
      CALL DISABLE
C     READ THTA,THTAD,THTADD
C          T500 T501  T502
C          A-17 A-15  A-13
    7 CALL ADL(0,THTA,THTAD,THTADD)
      IF (SENSE SWITCH 5) 77,78
   77 CALL HØLD
      PRINT 110,THTA,THTAD,THTADD,TH,THP,THF,IPHASE,ICØUNT
  110 FØRMAT( ' THTA =',F 7.4,'  THTAD =',F 7.4,'  THTADD =',F 7.4,
     1  '  IERRØR =',F7.4,'  THP=',F7.4,'  THF=',F7.4,'  IPHASE =',I2,
     2  '  TIME =',I4)
      CALL CØMPUTE
  178 IF (TEST(13) .GT. 0) GØ TØ 178
   78 ICØUNT=ICØUNT+1
      THF = -THTA
```

```
      IP=IPHASE+1
      GØ TØ (18,8,9,24,23,25),IP
C           PHASE ZERØ
   18 IF (THTA .GE. CHG) GØ TØ 32
      IF (THTADD .LT. S2  ) GØ TØ 80
      THP = .0
      IPHASE = 1
      IF (SENSE SWITCH 6) 160,80
  160 CALL HØLD
      PRINT 110,THTA,THTAD,THTADD,TH,THP,THF,IPHASE,ICØUNT
      CALL CØMPUTE
  170 IF (TEST(13) .GT. 0) GØ TØ 170
      GØ TØ 80
C           PHASE ØNE
    8 IF (THTA .GE. CHG) GØ TØ 32
      IF (THTADD .GT. .30) GØ TØ 80
      IF(-THTAD .GE. .6 ) GØ  TØ 31
      IF (THTA.GT. SW3) GØ TØ 31
      IPHASE = 0
      THP = -K2
      IF (SENSE SWITCH 6) 161,80
  161 CALL HØLD
      PRINT 110,THTA,THTAD,THTADD,TH,THP,THF,IPHASE,ICØUNT
      CALL CØMPUTE
  171 IF (TEST(13) .GT. 0) GØ TØ 171
      GØ TØ 80
   31 THP =0
      IPHASE = 2
      IF (SENSE SWITCH 6) 162,88
  162 CALL HØLD
      PRINT 110,THTA,THTAD,THTADD,TH,THP,THF,IPHASE,ICØUNT
      CALL CØMPUTE
  172 IF (TEST(13) .GT. 0) GØ TØ 172
   88 S2  = SW2 / 2.
      IF (IFLP .EQ. 0) GØ TØ 80
```

```
      CHG = THR -      THTA
      IFLP = 0
      GO TO 80
C          PHASE TWO
    9 IF(-THTAD .GE. .5 ) GO  TO 32
      IF (THTA.GT. SW4) GO TO 32
      IPHASE = 0
      THP = -K2
      IF (SENSE SWITCH 6) 163,80
  163 CALL HOLD
      PRINT 110,THTA,THTAD,THTADD,TH,THP,THF,IPHASE,ICOUNT
      CALL COMPUTE
  173 IF (TEST(13) .GT. 0) GO TO 173
      GO TO 80
   32 THP = 0
      IF (THTA.LT.  CHG) GO TO 80
      THP = K2
      IPHASE = 3
      S2 = THTAD
      IF (SENSE SWITCH 6) 164,80
  164 CALL HOLD
      PRINT 110,THTA,THTAD,THTADD,TH,THP,THF,IPHASE,ICOUNT
      CALL COMPUTE
  174 IF (TEST(13) .GT. 0) GO TO 174
      GO TO 80
C          PHASE THREE
   24 IF ( THTADD.GT.  S2 ) GO TO 80
      THP = -K2
      IPHASE = 4
      IF (SENSE SWITCH 6) 165,80
  165 CALL HOLD
      PRINT 110,THTA,THTAD,THTADD,TH,THP,THF,IPHASE,ICOUNT
      CALL COMPUTE
  175 IF (TEST(13) .GT. 0) GO TO 175
      GO TO 80
```

```
C           PHASE FØUR
   23 IF ( THTADD .LT.-.40)GØ TØ 80
      THP = -K2/2.
      IF ( THTADD .LT.-.1) GØ TØ 80
C           PHASE FIVE
   25 THP = 0
      THF = -THR
      IPHASE = 5
      IF (SENSE SWITCH 6) 166,80
  166 CALL HØLD
      PRINT 110,THTA,THTAD,THTADD,TH,THP,THF,IPHASE,ICØUNT
      CALL CØMPUTE
  176 IF (TEST(13) .GT. 0) GØ TØ 176
   80 TH = ABS(THR-THTA)
      IF (TH .LE. E) GØ TØ 90
      IF (THTA .LT. SW3) GØ TØ 81
      IF (IPHASE .NE. 2) GØ TØ 81
      IF (THTA .GT. CHG) GØ TØ 32
   81 IF (SENSE SWITCH 1) 10,11
   11 IF (SENSE SWITCH 2) 20,12
   12 IF (SENSE SWITCH 3) 30,13
   13 IF (SENSE SWITCH 4) 40,14
   14 IF (IREP.LE.ICØUNT) GØ TØ 90
      CALL DAC(1,-THP,2,THF)
      GØ TØ 7
   10 ØUTPUT(101)'IREP'
      INPUT(101)
      GØ TØ 5
   20 ØUTPUT(101)'THR'
      INPUT(101)
      GØ TØ 5
   30 ØUTPUT(101)'K1'
      INPUT(101)
      GØ TØ 5
   40 ØUTPUT(101)'K2'
```

```
      INPUT(101)
      GO TO 5
   90 THP = 0
      THF = -THR
      CALL DAC(1,-THP,2,THF)
      DO 91 I = 1,200000
   91 ALOOP = 1. * 0.
      PRINT 111,TH,THR,CHG,SW3,SW4
  111 FORMAT(10X,'TH =',F10.6,'    THR = ',F10.6,'     CHG =',F10.6,
     1  '   SW3 =',F10.6,'     SW4 =',F10.6)
      CALL HOLD
      CALL SCAN(4HA017,THTA ,4HA015,THTAD,4HA013,THTADD)
      PRINT 110,THTA,THTAD,THTADD,TH,THP,THF,IPHASE,ICOUNT
      THI = THP
      GO TO 1
      END
C
C***********************************************************************
C
C
C         THIS SUBROUTINE SETS A FLAG UPON AN ANALOG INTERRUPT
C
C
C***********************************************************************
C
C
      RECURSIVE SUBROUTINE ENDSTP(IFL)
      LOGICAL IFL
      IFL= .FALSE.
      RETURN
      END
C
.1000   .1000   .3000   .1000   .3000   .6000   .5000   .6000   .9999
.3000   .3000   .3333   .0666   .1000   .1000   .1000   .1000   .1000
```


Man
Dig
T421
Digital
T420
E_{AF}
$-\dot{E}_{GF}/10$
$E_{GF}/10$
$-2\dot{I}_{GF}$
$-I_A/10$
$10\ddot{\theta}$
$-3\dot{\theta}$
$\theta_M/5$
.1
.1
.3
.3
.1
.3
.6
.5
.6
.9999
.3333
.3
.3
.3333
.0666
.1

LIST OF REFERENCES

1. C. T. Apple, The Program Monitor - A Device for Program Performance Measurements, ACM 20th National Conference, 1965.

2. F. D. Schulman, Hardware Measurement Device for IBM System 360 Time Sharing Evaluation, ACM 22nd National Conf., 1967.

3. J. J. Cohn, L. A. Whitaker, Improved Techniques in Diagnostic Programming, Sylvania Technologist, Vol 3, p. 90-96, July 1960.

4. D. J. Farthing, A Fail-Safe Strategy for Commerical Processors, IEEE Conference Pub #55, Computer Group, England, 1969.

5. K. Maling, E. L. Allen, Jr., A Computer Organization and Programming System for Automated Maintenance, IEEE Trans. on Electronic Computers, Vol EC-12, p. 887-895, December 1963.

6. W. F. Breuer, Design Automation of Digital Computers, Proceedings of the IEEE, December 1966.

7. J. P. Eckert, Checking Circuits and Diagnostic Routines, IRE National Conv. Rec., pt. 7, p. 62-65, 1953.

8. H. Y. Chang, An Algorithm for Selecting an Optimum Set of Diagnostic Tests, IEEE Transactions on Electronic Computers, Vol EC-14, p. 706-711, October 1965.

9. E. Manning, On Computer Self-Diagnosis, IEEE Transactions on Electronic Computers EC-15, Nov. 6, p. 873-890, December 1966.

10. R. E. Forbes, D. H. Rutherford, C. B. Stieglitz, L. H. Tung, A Self-Diagnosible Computer, Proc. FJCC, Vol. 27, p. 1073-1086, 1965 and IBM rept 65-825-1486.

11. W. R. McCormack, C. Michel, Diagnostic Maintenance: A Technique Using a Computer, IEEE Trans, on Aerospace, Vol AS-1, p. 931-941, August 1963.

12. A. L. Scherr, Time-Sharing Measurement, Datamation, p. 22-26, April 1966.

13. J. T. Quatse, Time Sharing Troubleshooter Repairs Computers On-Line, Electronics, January 24, 1966.

14. A. Avizienis, Design of Fault-Tolerant Computers, FJCC; p. 733-743, 1967.

15. W. C. Carter, H. C. Montgomery, R. J. Priss, H. J. Reinheimer, Design of Serviceability Features of the IBM System/360, IBM Journal, p. 115-126, April 1964.

16. E. J. Kletsky, Diagnosis of Equipment Failures, Rome Air Development Center, Griffis AFB, N. Y., Tech. Rept 60-67B, April 1960.

17. T. R. Baskow, J. Friets, A. Harson, A Programming System for Detection and Diagnosis of Machine Malfunction, IEEE Transaction on Electronic Computers, p. 10-17, February 1963.

18. H. N. Cantrell, A. L. Ellison, Multiprogramming System Performance Measurement and Analysis, SJCC, 1968.

BIBLIOGRAPHY

A. COMPLETE LIST OF ARTICLES

1. P. W. Agnew, R. E. Forbes, C. B. Stieglity, An Approach to Self-Repairing Computers, Paper No. 6.3 Conference Digest of the 1st Annual IEEE Computer Conference, Chicago, Illinois, September 6-8, 1967.

2. J. E. Anderson, F. J. Macri, Multiple Redundancy Applications in a Computer, Proc. 1967 Annual Symposium on Reliability, 553-562, Washington, D. C., January 10-12, 1967.

3. C. T. Apple, The Program Monitor - A Device for Program Performance Measurements, ACM 20th National Conference, 1965.

4. A. Avizienis, Design of Fault-Tolerant Computers, FJCC, p. 733-743, 1967.

5. A. Avizienis, Concurrent Diagnosis of Arithmetic Processors, Paper No. 3.1, Conference Digest of the 1st Annual IEEE Computer Conference, Chicago, Illinois, September 6-8, 1967.

6. A. Avizienis, The Diagnosable Arithmetic Processor, JPL Space Programs Summary 37-37 IV, Jet Propulsion Laboratory, Pasadena, Calif., p. 76-80, 1966.

7. N. A. Ball, H. Q. Foster, W. H. Long, I. E. Sutherland, R. L. Wigington, A Shared Memory Computer Display System, IEEE Transactions on Electronic Computers, Vol EC-15, p. 750, October 1966.

8. T. R. Baskow, J. Friets, A. Harson, A Programming System for Detection and Diagnosis of Machine Malfunction, IEEE Transaction on Electronic Computers, p. 10-17, February 1963.

9. R. V. Bock, A. P. Toth, Hardware and Software for Maintenance in the B5500 processor, IEEE International Convention, New York, N. Y., 1965.

10. W. G. Bouricius, W. C. Carter, J. P. Roth, P. R. Schneider, Investigations in the Design of an Automatically Repaired Computer, Paper 6.4 Conference Digest of the 1st Annual IEEE Computer Conference, Chicago, Illinois, September 6-8, 1967.

11. W. F. Breuer, Design Automation of Digital Computers, Proceedings of the IEEE, December 1966.

12. P. Calingaert, System Performance Evaluation: Survey and Appraisal, Comm. of ACM Vol 10 No. 1, p. 12-18, January 1967.

13. H. N. Cantrell, A. L. Ellison, Multiprogramming System Performance Measurement and Analysis, SJCC, 1968.

14. W. C. Carter, H. C. Montgomery, R. J. Priss, H. J. Reinheimer, Design of Serviceability Features of the IBM System 360, IBM Journal, p. 115-126, April 1964.

15. H. Y. Chang, An Algorithm for Selecting an Optimum Set of Diagnostic Tests, IEEE Transactions on Electronic Computers, Vol EC-14 p. 706-711, October 1965.

16. S. H. Chasen, The Introduction of Man-Computer Graphics into the Aerospace Industry, FJCC proc. Vol 27 p. 883-891, 1965.

17. R. L. Chew, Note on Timing Simulation of a Large Asynchronous Computer, Computer Journal Vol 7 No 2, p. 122-123, July 1964.

18. W. Chu, Diagnosis of System Failures, IBM Tech. Rept. TR 02.294, March 26, 1964.

19. J. J. Cohn, L. A. Whitaker, Improved Techniques in Diagnostic Programming, Sylvania Technologist, Vol 3, p. 90-96, July 1960.

20. W. T. Comfort, A Computing System Design for User Service, Proc. Fall Joint Computer Conference p. 619-626, 1965.

21. J. E. Dammann, E. J. Skiko, E. V. Weber, A Data Display Subsystem, IBM Journal of R & D, Vol 7, No 4, p. 325, October 1963.

22. W. R. Deniston, Sipe! A TSS/360 Software Measurement Technique, Proc of 24th National Conference ACM, 1969.

23. D. Emuller, Evaluation of Logical and Organizational Methods for Improving the Reliability and Availability of a Computer, Paper No. 6.1, Conference Digest of the 1st Annual IEEE Computer Conference, Chicago, Illinois, September 6-8, 1967.

24. H. L. Ergott, D. P. Rosenberg, Modular Redundancy for Spaceborne Computers, Proc of IEEE Spaceborne Computer Engineering Conference, p. 137-150, Anaheim, California, October 30-31, 1962.

25. G. Estrin, D. Hopkins, B. Coggan, S. D. Crocker, Snuper Computer, a Computer in Instrumentation Automation, SJCC, p. 645-656, 1967.

26. D. J. Farthing, A Fail-Safe Strategy for Commercial Processors, IEEE Conference Pub #55, Computer Group, England, 1969.

27. S. I. Firstman, B. Gluss, Search Rules for Automatic Fault Location, The Rand Corp, Santa Monica, Calif, U.S. Air Force Project, Res. Memo RM-2514, June 1960.

28. R. E. Forbes, C. B. Stieglitz, D. Muller, Automatic Fault Diagnosis, presented at the AIEE Conf. on Diagnosis of Failures in Switching Circuits, 1961.

29. R. E. Forbes, D. H. Rutherford, C. B. Stieglitz, L. H. Tung, A Self-Diagnosible Computer, Proc. FJCC, Vol 27, p. 1073-1086, 1965 and IBM Rept. 65-825-1486.

30. R. L. Fulton, Programming Graphic Devices, Computers and Automation, Vol. 14, p. 17-19, November 1965.

31. R. L. Fulton, Visual Input to Computers, Datamation p. 37-40, November 1963.

32. B. Gluss, An Optimum Policy for Detecting a Fault in a Complex System, J. Operations Research Soc. Am., Vol 7, July-August 1959.

33. F. J. Hackl, R. W. Shirk, An Integrated Approach to Automated Computer Maintenance, 1965 Sixth Annual Symp. on Switching Circuit Theory and Logical Design, IEEE Special Publ, 16C13 p. 289-302, 1965.

34. D. J. Herman, F. C. Ihrer, The Use of a Computer to Evaluate Computers, Proc SJCC, Spartan Books, Washington, D. C., p. 383-395, 1964.

35. D. J. Herman, SCER! a Computer Evaluation Tool, Datamation Vol 13, No 2, p. 26-28, February 1967.

36. E. S. Hoover, B. J. Eckhart, Performance of a Monitor for a Real-Time Control System, Proc FJCC Spartan Books, Washington, D. C. p. 23-35, 1966.

37. J. H. Katz, Simulation of a Multiprocessor Computer System, Proc SJCC Spartan Books, Washington, D. C., p. 127-139, 1966.

38. E. J. Kletsky, Diagnosis of Equipment Failures, Rome Air Development Center, Griffis AFB, N. Y. Tech. Rept. 60-67B, April 1960.

39. J. K. Knox-Seith, Improving the Reliability of Digital Systems by Redundancy and Restoring organs, PhD thesis, Electrical Engineering, Stanford University, August 1964.

40. B. Krishnamoorthi, R. C. Wood, Time-Shared Computer Operations with Both Inter-Arrival and Service Times Exponential, System Development Corporation SP1848 October 1964.

41. B. W. Lampson, Interactive Machine Language Programming, AFIPS Conference proceedings, Vol 27, part 1, 1965 FJCC, Spartan Books, Washington, D. C., p. 473-481, 1965.

42. R. S. Ledley, J. Jacobsen, M. Belson, Bugsys: a Programming System for Picture Processing - not for Debugging, Comm ACM, Vol 9, No 2, p. 79-84, February 1966.

43. C. Machover, Converting Data to Human-Interpretable Form, Data System Design, p. 17-23, September 1964.

44. K. Maling, Classes of Component Failures and a System Design to Facilitate Their Location, IBM Data Systems Div. Tech. Rept., TR00.1029, 1964.

45. K. Maling, E. L. Allen, Jr., A Computer Organization and Programming System for Automated Maintenance, IEEE Trans. on Electronic Computers, Vol EC-12, p. 887-895, December 1963.

46. E. G. Manning, Self-Diagnosis of Electronic Computers - An Experimental Study, Coordinated Science Lab. Report R-259, University of Illinois, Urbana, July 1965.

47. E. Manning, On Computer Self-Diagnosis, IEEE Trans. on Electronic Computers EC-15, No 6, p. 873-890, December 1966.

48. F. P. Mathur, A. Avizienis, Reliability Analysis and Architecture of a Hybrid-Redundant Digital System: Generalized Triple Modular Redundancy with Self-Repair, SJCC, p. 375-383, 1970.

49. W. R. McCormack, C. Michel, Diagnostic Maintenance: A Technique Using a Computer, IEEE Trans. on Aerospace, Vol AS-1, p. 931-941, August 1963.

50. N. R. Nielsen, The Analysis of General Purpose Computer Time-Sharing Systems, Stanford Computation Center Document 40-10-1 December 1966.

51. W. R. Pfeil, N. H. Quan, Self-Testing and Verification Techniques as Applied to an Airborne Military Guidance Computer, IRE Trans. On Military Electronics, Vol MIL-6, p. 268-275, July 1962.

52. W. H. Pierce, Failure - Tolerant Computer Design, Academic Press, Inc., New York, 1965.

53. J. F. Poage, The Derivation of Optimum Tests for Logic Circuits, PhD dissertation, Princeton University, January 1963.

54. J. T. Quatse, Time Sharing Troubleshooter Repairs Computers On-Line, Electronics, January 24, 1966.

55. D. J. Roek, W. C. Emerson, A Hardware Instrumentation Approach to Evaluation of a Large Scale System, ACM 24th National Meeting, p. 69, 1969.

56. J. Rosa, Command and Control Display Technology Since 1962, Information System Science, Proc. of the second congress, Hot Springs Va, Sparton Books, Washington, D. C. p. 411-414, 22-25 November 1964.

57. A. B. Salisbury, P. H. Enslow, Jr., Diagnostic Programming for Digital Computers! A Bibliography, Tech. Report, ELEC-67-1, U. S. Military Academy, West Point, N. Y., April 1967.

58. A. L. Scherr, An Analysis of Time-Shared Computer Systems, Massachusetts Institute of Technology, MAC-TR-18, Project MAC, June 1965.

59. A. L. Scherr, Time-Sharing Measurement, Datamation, p. 22-26, April 1966.

60. F. D. Schulman, Hardware Measurement Device for IBM System 360 Time Sharing Evaluation, ACM 22nd National Conference, 1967.

61. S. Seshu, On an Improved Diagnostic Program, IEEE Trans, on Electronic Computers, Vol EC-13, p. 76-79, February 1965.

62. A. V. Shortell, Jr., On-Line Programming, Datamation, Vol 11, No 1, p. 29-30, January 1965.

63. N. Statland, Methods for Evaluating Computer Systems Performance, Computers and Automation, Vol 13, No 2, p. 18-23, February 1964.

64. T. G. Stockham, Jr., Some Methods of Graphical Debugging, IBM Scientific Computing Symposium, Man-Machine Communications, Thomas J. Watson Research Center, Yorktown Heights, New York May 3-5, 1965.

65. I. E. Sutherland, The Future of On-Line Systems, On-Line Computing Systems, American Data Processing, Inc., Detroit, Mich. p. 9-13, 1965.

66. I. E. Sutherland, Computer Graphics: Ten Unsolved Problems, Datamation, Vol 12, No 5, p. 22-27, May 1966.

67. R. Toeste, Design of a Repairable Redundant Computer, IRE Trans on Electronic Computers EC-11, No 5, p. 643-649, October 1962.

68. I. Terris, M. A. Melkanoff, Investigating and Simulation of a Self-Repairable Digital Computer, IEEE Conference Record on Switching Circuit Theory and Logical Design, p. 264-272, 1965.

69. R. L. Walquist, The Isolation and Correction of Computer Malfunctions, U. S. Government Rept AD 609276, September 1965.

70. A. Van Dam, Computer Driven Displays, and Their Uses in Man/Machine Interaction, Advances in Computers, Vol VII, Academic Press, 1966.

71. C. Walter, Graphical Data Monitoring With Computer Controlled Color Oscilloscopes, Proc of the Filmed Data & Computers Symposium, Society of Photo-Optical Instrumentation Engineers, 13-14 June 1966.

B. SOFTWARE RELATED ARTICLES CROSS-REFERENCE

	Methods	Reference
1.	Basic Fault Detection and Diagnostics	4, 8, 11, 14, 19, 26, 29, 45, 47, 51
2.	Monitoring Programs by Inserting Calls into the Programs	22, 25
3.	Monitors Specific Registers or Data Points and Dumps the Data onto Tape	14, 55
4.	Monitors Scheduling Algorithm for Later Analysis	13, 14, 59
5.	Basic Fault Detection and Diagnosis but Partitions the Computer into Sections	26, 29
6.	Basic Fault Detection and Diagnosis but uses Matrix Manipulations to Locate Fault and Erring Part	49
7.	Fault Detection Only, Manual Diagnosis	19, 51
8.	Fault Detection and Diagnosis by Simulation Only	61
9.	Reorganizes System Upon a Detected Fault	4, 14, 26
10.	On Line Requests for Maintenance Routines	14, 54

C. HARDWARE METHODS USED

	Method	References
1.	Voting Circuits	2, 4, 48, 67
2.	Monitors Registers or Signals into Counters and Dumps Data onto Tape	3, 55, 60
3.	Special Purpose Fault Detectors	
	a. Hardware only	2, 4, 19
	b. Hardware with program settable and accessable data	11, 14, 26, 45

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Naval Tactical Data System, Code 6172 Naval Ship Engineering Center Hyattsville, Maryland 20782	1
4.	Associate Professor G. A. Rahe, Code 52 Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
5.	Professor Robert B. McGhee Department of Electrical Engineering and Computer Information Science Ohio State University Columbus, Ohio 43210	1
6.	Assistant Professor F. W. Terman, Code 52 Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
7.	Instructor LT R. D. DeLaura, USNR, Code 52 Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
8.	LT E. A. Potter, USN 12404 Rockledge Drive Bowie, Maryland 20715	1

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)
Naval Postgraduate School
Monterey, California 93940

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP

3. REPORT TITLE
A Technique for Monitoring a Real-Time Multi-Computer System Under Heavy Loading

4. DESCRIPTIVE NOTES (Type of report and, inclusive dates)
Master's Thesis; December 1970

5. AUTHOR(S) (First name, middle initial, last name)
Edward Allen Potter

6. REPORT DATE
December 1970

7a. TOTAL NO. OF PAGES
164

7b. NO. OF REFS
18

8a. CONTRACT OR GRANT NO.

b. PROJECT NO.

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S)

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT
This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY
Naval Postgraduate School
Monterey, California 93940

13. ABSTRACT

Techniques are developed for the design of a monitor of a real-time multi-computer system that is under heavy loading. The first portion relates to the requirements of partitioning to aid in fault recognition and diagnostic routines. The dynamic allocation of system time to the system tasks and fault monitoring is developed secondly. System reconfiguration of the partitioned subsystems restores the system to operation at a degraded level until faults are corrected. The paper discusses a Ship Combat Weapon System as an example of a large scale multi-computer system monitor.

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Fault monitoring						
Diagnostic routines						
Real-Time computer						
Dynamic time allocation						
Ship combat weapon system						
Multi-Computer system monitor						
Partitioning and reconfiguration						

Thesis
P7488 Potter
c.1 A technique for
monitoring a real-
time multi-computer
system under heavy
loading.